STATA
STRUCTURAL EQUATION MODELING
REFERENCE MANUAL
RELEASE 16

A Stata Press Publication
StataCorp LLC
College Station, Texas
## Contents

Acknowledgments ................................................................................................................. 1

Intro 1 ................................................................................................................................. 2
Intro 2 ................................................................................................................................. 7
Intro 3 ................................................................................................................................. 38
Intro 4 ................................................................................................................................. 45
Intro 5 ................................................................................................................................. 64
Intro 6 ................................................................................................................................. 87
Intro 7 ................................................................................................................................. 98
Intro 8 ................................................................................................................................. 107
Intro 9 ................................................................................................................................. 109
Intro 10 ............................................................................................................................... 113
Intro 11 ............................................................................................................................... 115
Intro 12 ............................................................................................................................... 123

Builder ............................................................................................................................... 133
Builder, generalized .......................................................................................................... 136

estat eform ......................................................................................................................... 139
estat eqgof ......................................................................................................................... 141
estat eqtest ......................................................................................................................... 143
estat framework ................................................................................................................ 145
estat ggof ........................................................................................................................... 147
estat ginvariant ................................................................................................................ 149
estat gof .............................................................................................................................. 151
estat lcgof .......................................................................................................................... 154
estat lcmean ....................................................................................................................... 156
estat lcprob ......................................................................................................................... 158
estat mindices ................................................................................................................... 160
estat residuals ................................................................................................................... 162
estat scoretests .................................................................................................................. 165
estat sd ............................................................................................................................... 167
estat stable ......................................................................................................................... 169
estat stdize ......................................................................................................................... 171
estat summarize ............................................................................................................... 173
estat teffects ...................................................................................................................... 174

Example 1 ............................................................................................................................ 177
Example 2 ............................................................................................................................ 185
Example 3 ............................................................................................................................ 190
Example 4 ............................................................................................................................ 198
Example 5 ............................................................................................................................ 201
Example 6 ............................................................................................................................ 204
Example 7 ............................................................................................................................ 208
Example 8 ............................................................................................................................ 216
Example 9 ............................................................................................................................ 220
Example 10 ........................................................................................................................... 229
Example 11 ........................................................................................................................... 236

SEM Builder ....................................................................................................................... 133
<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Seemingly unrelated regression</td>
<td>239</td>
</tr>
<tr>
<td>13</td>
<td>Equation-level Wald test</td>
<td>243</td>
</tr>
<tr>
<td>14</td>
<td>Predicted values</td>
<td>244</td>
</tr>
<tr>
<td>15</td>
<td>Higher-order CFA</td>
<td>246</td>
</tr>
<tr>
<td>16</td>
<td>Correlation</td>
<td>253</td>
</tr>
<tr>
<td>17</td>
<td>Correlated uniqueness model</td>
<td>258</td>
</tr>
<tr>
<td>18</td>
<td>Latent growth model</td>
<td>265</td>
</tr>
<tr>
<td>19</td>
<td>Creating multiple-group summary statistics data</td>
<td>272</td>
</tr>
<tr>
<td>20</td>
<td>Two-factor measurement model by group</td>
<td>277</td>
</tr>
<tr>
<td>21</td>
<td>Group-level goodness of fit</td>
<td>287</td>
</tr>
<tr>
<td>22</td>
<td>Testing parameter equality across groups</td>
<td>288</td>
</tr>
<tr>
<td>23</td>
<td>Specifying parameter constraints across groups</td>
<td>291</td>
</tr>
<tr>
<td>24</td>
<td>Reliability</td>
<td>297</td>
</tr>
<tr>
<td>25</td>
<td>Creating summary statistics data from raw data</td>
<td>301</td>
</tr>
<tr>
<td>26</td>
<td>Fitting a model with data missing at random</td>
<td>309</td>
</tr>
<tr>
<td>27g</td>
<td>Single-factor measurement model (generalized response)</td>
<td>313</td>
</tr>
<tr>
<td>28g</td>
<td>One-parameter logistic IRT (Rasch) model</td>
<td>320</td>
</tr>
<tr>
<td>29g</td>
<td>Two-parameter logistic IRT model</td>
<td>329</td>
</tr>
<tr>
<td>30g</td>
<td>Two-level measurement model (multilevel, generalized response)</td>
<td>339</td>
</tr>
<tr>
<td>31g</td>
<td>Two-factor measurement model (generalized response)</td>
<td>349</td>
</tr>
<tr>
<td>32g</td>
<td>Full structural equation model (generalized response)</td>
<td>356</td>
</tr>
<tr>
<td>33g</td>
<td>Logistic regression</td>
<td>363</td>
</tr>
<tr>
<td>34g</td>
<td>Combined models (generalized responses)</td>
<td>368</td>
</tr>
<tr>
<td>35g</td>
<td>Ordered probit and ordered logit</td>
<td>374</td>
</tr>
<tr>
<td>36g</td>
<td>MIMIC model (generalized response)</td>
<td>382</td>
</tr>
<tr>
<td>37g</td>
<td>Multinomial logistic regression</td>
<td>387</td>
</tr>
<tr>
<td>38g</td>
<td>Random-intercept and random-slope models (multilevel)</td>
<td>396</td>
</tr>
<tr>
<td>39g</td>
<td>Three-level model (multilevel, generalized response)</td>
<td>412</td>
</tr>
<tr>
<td>40g</td>
<td>Crossed models (multilevel)</td>
<td>420</td>
</tr>
<tr>
<td>41g</td>
<td>Two-level multinomial logistic regression (multilevel)</td>
<td>425</td>
</tr>
<tr>
<td>42g</td>
<td>One- and two-level mediation models (multilevel)</td>
<td>435</td>
</tr>
<tr>
<td>43g</td>
<td>Tobit regression</td>
<td>444</td>
</tr>
<tr>
<td>44g</td>
<td>Interval regression</td>
<td>447</td>
</tr>
<tr>
<td>45g</td>
<td>Heckman selection model</td>
<td>451</td>
</tr>
<tr>
<td>46g</td>
<td>Endogenous treatment-effects model</td>
<td>460</td>
</tr>
<tr>
<td>47g</td>
<td>Exponential survival model</td>
<td>467</td>
</tr>
<tr>
<td>48g</td>
<td>Loglogistic survival model with censored and truncated data</td>
<td>471</td>
</tr>
<tr>
<td>49g</td>
<td>Multiple-group Weibull survival model</td>
<td>477</td>
</tr>
<tr>
<td>50g</td>
<td>Latent class model</td>
<td>483</td>
</tr>
<tr>
<td>51g</td>
<td>Latent class goodness-of-fit statistics</td>
<td>489</td>
</tr>
<tr>
<td>52g</td>
<td>Latent profile model</td>
<td>492</td>
</tr>
<tr>
<td>53g</td>
<td>Finite mixture Poisson regression</td>
<td>501</td>
</tr>
<tr>
<td>54g</td>
<td>Finite mixture Poisson regression, multiple responses</td>
<td>507</td>
</tr>
</tbody>
</table>

**gsem** | Generalized structural equation model estimation command | 511  |
**gsem estimation options** | Options affecting estimation | 516  |
**gsem family-and-link options** | Family-and-link options | 521  |
**gsem group options** | Fitting models on different groups | 527  |
**gsem lclass options** | Fitting models with latent classes | 529  |
**gsem model description options** | Model description options | 531  |
**gsem path notation extensions** | Command syntax for path diagrams | 533  |
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>gsem postestimation</td>
<td>544</td>
</tr>
<tr>
<td>gsem reporting options</td>
<td>547</td>
</tr>
<tr>
<td>lincom</td>
<td>549</td>
</tr>
<tr>
<td>lrtest</td>
<td>550</td>
</tr>
<tr>
<td>Methods and formulas for gsem</td>
<td>552</td>
</tr>
<tr>
<td>Methods and formulas for sem</td>
<td>572</td>
</tr>
<tr>
<td>nlcom</td>
<td>584</td>
</tr>
<tr>
<td>predict after gsem</td>
<td>586</td>
</tr>
<tr>
<td>predict after sem</td>
<td>591</td>
</tr>
<tr>
<td>sem</td>
<td>593</td>
</tr>
<tr>
<td>sem and gsem option constraints()</td>
<td>598</td>
</tr>
<tr>
<td>sem and gsem option covstructure()</td>
<td>600</td>
</tr>
<tr>
<td>sem and gsem option from()</td>
<td>603</td>
</tr>
<tr>
<td>sem and gsem option reliability()</td>
<td>606</td>
</tr>
<tr>
<td>sem and gsem path notation</td>
<td>609</td>
</tr>
<tr>
<td>sem and gsem syntax options</td>
<td>615</td>
</tr>
<tr>
<td>sem estimation options</td>
<td>616</td>
</tr>
<tr>
<td>sem group options</td>
<td>618</td>
</tr>
<tr>
<td>sem model description options</td>
<td>620</td>
</tr>
<tr>
<td>sem option method()</td>
<td>622</td>
</tr>
<tr>
<td>sem option noxconditional</td>
<td>624</td>
</tr>
<tr>
<td>sem option select()</td>
<td>627</td>
</tr>
<tr>
<td>sem path notation extensions</td>
<td>629</td>
</tr>
<tr>
<td>sem postestimation</td>
<td>633</td>
</tr>
<tr>
<td>sem reporting options</td>
<td>635</td>
</tr>
<tr>
<td>sem ssd options</td>
<td>637</td>
</tr>
<tr>
<td>ssd</td>
<td>639</td>
</tr>
<tr>
<td>test</td>
<td>643</td>
</tr>
<tr>
<td>testnl</td>
<td>645</td>
</tr>
<tr>
<td>Glossary</td>
<td>647</td>
</tr>
<tr>
<td>Subject and author index</td>
<td>659</td>
</tr>
</tbody>
</table>
Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals, for example, [U] 27 Overview of Stata estimation commands; [R] regress; and [D] reshape. The first example is a reference to chapter 27, Overview of Stata estimation commands, in the User’s Guide; the second is a reference to the regress entry in the Base Reference Manual; and the third is a reference to the reshape entry in the Data Management Reference Manual.

All the manuals in the Stata Documentation have a shorthand notation:

- [GSM] Getting Started with Stata for Mac
- [GSU] Getting Started with Stata for Unix
- [GSW] Getting Started with Stata for Windows
- [U] Stata User’s Guide
- [R] Stata Base Reference Manual
- [BAYES] Stata Bayesian Analysis Reference Manual
- [D] Stata Data Management Reference Manual
- [FN] Stata Functions Reference Manual
- [LASSO] Stata Lasso Reference Manual
- [XT] Stata Longitudinal-Data/Panel-Data Reference Manual
- [MI] Stata Multiple-Imputation Reference Manual
- [RPT] Stata Reporting Reference Manual
- [SVY] Stata Survey Data Reference Manual
- [I] Stata Glossary and Index
sem and gsem were developed by StataCorp. Neither command would exist without the help of two people outside of StataCorp. We must thank these two people profusely. They are

Jeroen Weesie, Department of Sociology at Utrecht University, The Netherlands
Sophia Rabe-Hesketh, University of California, Berkeley

Jeroen Weesie is responsible for the existence of the SEM project at StataCorp. While spending his sabbatical with us, Jeroen expressed—repeatedly—the importance of SEM, and that enthusiasm for SEM was disregarded—repeatedly. Not until after his sabbatical did StataCorp see the light. At that point, we had him back, and back, and back, so that he could inspire us, guide us, tell us what we had right, and, often, tell us what we had wrong.

Jeroen helped us with the math, the syntax, and system design, and, when we were too thick-headed, he even wrote code. By the date of first shipment, all code had been rewritten by us, but design and syntax for SEM still now and forever will show Jeroen’s influence.

Thank you, Jeroen Weesie, for teaching us SEM.

Sophia Rabe-Hesketh contributed a bit later, after the second project, GSEM, was well underway. GSEM stands for generalized SEM. Sophia is the coauthor of gllamm and knows as much about multilevel and structural equation modeling as anybody, and probably more. She helped us a lot through her prolific published works; we did have her visit a few times, though, mainly because we knew that features in GSEM would overlap with features in GLLAMM, and we wanted to straighten out any difficulties that competing features might cause.

About the competing features, Sophia cared nothing. About the GSEM project, she was excited. About syntax and computational methods—well, she straightened us out the first day, even on things we thought we had settled. Today, enough of the underlying workings of GSEM are based on Sophia’s and her coauthors’ publications that anyone who uses gsem should cite Rabe-Hesketh, Skrondal, and Pickles (2004).

We are indebted to the works of Sophia Rabe-Hesketh, Anders Skrondal of the University of Oslo and the Norwegian Institute of Public Health, and Andrew Pickles of the University of Manchester.

Reference

Also see
[R] gllamm — Generalized linear and latent mixed models
Description

SEM stands for structural equation model. Structural equation modeling is

1. A notation for specifying SEMs.
2. A way of thinking about SEMs.
3. Methods for estimating the parameters of SEMs.

Stata’s \texttt{sem} and \texttt{gsem} commands fit these models: \texttt{sem} fits standard linear SEMs, and \texttt{gsem} fits generalized SEMs.

In \texttt{sem}, responses are continuous and models are linear regression.

In \texttt{gsem}, responses are continuous or binary, ordinal, count, or multinomial. Models are linear regression, gamma regression, logit, probit, ordinal logit, ordinal probit, Poisson, negative binomial, multinomial logit, and more.

\texttt{sem} fits models to single-level data.

\texttt{gsem} fits models to single-level or multilevel data. Latent variables can be included at any level. \texttt{gsem} can fit models with mixed effects, including random effects such as unobserved effects within patient, nested effects such as unobserved effects within patient within doctor, and crossed effects such as unobserved effects within occupation and country.

In \texttt{sem}, all latent variables are continuous.

In \texttt{gsem}, latent variables are continuous or categorical. A model can have continuous latent variables or categorical latent variables but not both. With categorical latent variables, \texttt{gsem} can fit latent class models and finite mixture models.

Meanwhile, \texttt{sem} provides features not provided by \texttt{gsem}: estimation using observations with missing values under the assumption of joint normality; goodness-of-fit statistics, modification indices, tests of indirect effects, and more; and models fit using summary-statistic data.

There is obviously overlap between the capabilities of \texttt{sem} and \texttt{gsem}. In such cases, results will be nearly equal. Results should be exactly equal because both commands are producing estimates of the same mathematical model, but \texttt{sem} and \texttt{gsem} use different numerical machinery. \texttt{sem}'s machinery requires less calculation and fewer approximations and so is faster and slightly more accurate.

Remarks and examples

Structural equation modeling encompasses a broad array of models from linear regression to measurement models to simultaneous equations, including along the way confirmatory factor analysis (CFA), correlated uniqueness models, latent growth models, multiple indicators and multiple causes (MIMIC) models, and item response theory (IRT) models.

Structural equation modeling is not just an estimation method for a particular model in the way that Stata’s \texttt{regress} and \texttt{probit} commands are, or even in the way that \texttt{stcox} and \texttt{mixed} are. Structural equation modeling is a way of thinking, a way of writing, and a way of estimating.
If you read the introductory manual pages in the front of this manual—[SEM] Intro 2, [SEM] Intro 3, and so on—we will do our best to familiarize you with SEM and our implementation of it.

Beginning with [SEM] Intro 2, entitled Learning the language: Path diagrams and command language, you will learn that

1. A particular SEM is usually described using a path diagram.
2. The \texttt{sem} and \texttt{gsem} commands allow you to use path diagrams to input models. In fact, \texttt{sem} and \texttt{gsem} share the same GUI, called the SEM Builder.
3. \texttt{sem} and \texttt{gsem} alternatively allow you to use a command language to input models. The command language is similar to the path diagrams.

[SEM] Intro 3, entitled Learning the language: Factor-variable notation (\texttt{gsem} only), amounts to a continuation of [SEM] Intro 2.

4. We teach you Stata’s factor-variable notation, a wonderfully convenient shorthand for including categorical variables in models.

In [SEM] Intro 4, entitled Substantive concepts, you will learn that

5. \texttt{sem} provides four different estimation methods; you need to specify the method appropriate for the assumptions you are willing to make. For \texttt{gsem}, there are two estimation methods.
6. There are four types of variables in SEMs: A variable is observed or latent, and simultaneously it is endogenous or exogenous. To this, \texttt{sem} and \texttt{gsem} add another type of variable, the error variable. Error variables are latent exogenous variables with a fixed-unit path coefficient, and they are associated with a single endogenous variable. Error variables are denoted with an \texttt{e}. prefix, so if \texttt{y1} is an endogenous variable, then \texttt{e.y1} is the associated error variable.
7. It is easy to specify path constraints in SEMs—you just draw them, or omit drawing them, on the diagram. It is similarly easy with the SEM Builder as well as with \texttt{sem}’s and \texttt{gsem}’s command language.
8. Determining whether an SEM is identified can be difficult. We show you how to let the software check for you.
9. Identification also includes normalization constraints. \texttt{sem} and \texttt{gsem} apply normalization constraints automatically, but you can control that if you wish. Sometimes you might even need to control it.

In [SEM] Intro 5, entitled Tour of models,

10. We take you on a whirlwind tour of some of the models that \texttt{sem} and \texttt{gsem} can fit. This is a fun and useful section because we give you an overview without getting lost in the details.

Then in [SEM] Intro 6, entitled Comparing groups,

11. We show you a highlight of \texttt{sem} and \texttt{gsem}: their ability to take an SEM and data consisting of groups—sexes, age categories, and the like—and fit the model in an interacted way that makes it easy for you to test whether and how the groups differ.

In [SEM] Intro 7, entitled Postestimation tests and predictions,

12. We show you how to redisplay results (\texttt{sem} and \texttt{gsem}), how to obtain exponentiated coefficients (\texttt{gsem} only), and how to obtain standardized results (\texttt{sem} only).
13. We show you how to obtain goodness-of-fit statistics (\texttt{sem} only).
14. We show you how to perform hypothesis tests, including tests for omitted paths, tests for relaxing constraints, and tests for model simplification.
15. We show you how to display other results, statistics, and tests.
16. We show you how to obtain predictions of observed response variables and predictions of latent variables. With \texttt{gsem}, you can obtain predicted means, probabilities, or counts that take into account the predictions of the latent variables, or you can set the latent variables to 0.

17. We show you how to access stored results.

In \texttt{SEM Intro 8}, entitled \textit{Robust and clustered standard errors},

18. We mention that \texttt{sem} and \texttt{gsem} optionally provide robust standard errors and provide clustered standard errors, which relaxes the assumption of independence of observations (or subjects) to independence within clusters of observations (subjects).

In \texttt{SEM Intro 9}, entitled \textit{Standard errors, the full story},

19. We provide lots of technical detail expanding on item 18.

In \texttt{SEM Intro 10}, entitled \textit{Fitting models with survey data},

20. We explain how \texttt{sem} and \texttt{gsem} can be used with Stata’s \texttt{svy:} prefix to obtain results adjusted for complex survey designs, including clustered sampling and stratification.

In \texttt{SEM Intro 11}, entitled \textit{Fitting models with summary statistics data (sem only)},

21. We show you how to use \texttt{sem} with summary statistics data such as the correlation or covariance matrix rather than the raw data. Many sources, especially textbooks, publish data in summary statistics form.

Finally, in \texttt{SEM Intro 12}, entitled \textit{Convergence problems and how to solve them},

22. We regretfully inform you that some SEMs have difficulty converging. We figure 5% to 15% of complicated models will cause difficulty. We show you what to do and it is not difficult.

In the meantime,

23. There are many examples that we have collected for you in \texttt{SEM Example 1}, \texttt{SEM Example 2}, and so on. It is entertaining and informative simply to read the examples in order.

24. There is an alphabetical glossary in \texttt{SEM Glossary}, located at the end of the manual.

If you prefer, you can skip all of this introductory material and go for the details. For the full experience, go directly to \texttt{SEM sem} and \texttt{SEM gsem}. You will have no idea what we are talking about—we promise.
The technical sections, in logical order, are

**Estimation**

- `sem`
- `gsem`
- `sem` and `gsem` path notation
- `sem` path notation extensions
- `gsem` path notation extensions
- `Builder`
- `Builder, generalized`
- `sem` model description options
- `gsem` model description options
- `sem` group options
- `gsem` group options
- `gsem` lclass options
- `sem` ssd options
- `sem` estimation options
- `gsem` estimation options
- `sem` reporting options
- `gsem` reporting options
- `sem` and `gsem` syntax options
- `sem` option noxconditional
- `sem` option select( )
- `sem` and `gsem` option covstructure( )
- `sem` option method( )
- `sem` and `gsem` option reliability( )
- `sem` and `gsem` option from( )
- `sem` and `gsem` option constraints( )
- `gsem` family-and-link options
- `ssd` (`sem` only)

**Postestimation, summary of**

- `sem` postestimation
- `gsem` postestimation

**Reporting results**

- `[R] estat`
- `estat summarize`
- `estat eform` (`gsem` only)
- `estat teffects` (`sem` only)
- `estat residuals` (`sem` only)
- `estat framework` (`sem` only)
- `estat sd` (`gsem` only)
- `estat lcmean` (`gsem` only)
- `estat lcprob` (`gsem` only)

**Goodness-of-fit tests**

- `estat gof` (`sem` only)
- `estat eqgof` (`sem` only)
- `estat ggof` (`sem` only)
- `estat lcgof` (`gsem` only)
- `[R] estat`
Hypotheses tests

- `estat mindices (sem only)`
- `estat eqtest (sem only)`
- `estat scoretests (sem only)`
- `estat ginvartant (sem only)`
- `estat stable (sem only)`
- `test`
- `lrtest`
- `testnl`
- `estat stdize (sem only)`

Linear and nonlinear combinations of results

- `lincom`
- `nlcom`

Predicted values

- `predict after sem`
- `predict after gsem`

Methods and formulas

- `Methods and formulas for sem`
- `Methods and formulas for gsem`

Many of these sections are technical, but mostly in the computer sense of the word. We suggest that when you read the technical sections, you skip to Remarks and examples. If you read the introductory sections, you will already know how to use the commands, so there is little reason to confuse yourself with syntax diagrams that are more precise than they are enlightening. However, the syntax diagrams do serve as useful reminders.

Also see

- `Intro 2 — Learning the language: Path diagrams and command language`
- `Example 1 — Single-factor measurement model`
- `Acknowledgments`
Description

Individual structural equation models are usually described using path diagrams. Path diagrams are described here.

Path diagrams can be used in sem’s (gsem’s) GUI, known as the SEM Builder or simply the Builder, as the input to describe the model to be fit. Path diagrams differ a little from author to author, and sem’s and gsem’s path diagrams differ a little, too. For instance, we omit drawing the variances and covariances between observed exogenous variables by default.

sem and gsem also provide a command-language interface. This interface is similar to path diagrams but is typable.

Remarks and examples

Remarks are presented under the following headings:

- Using path diagrams to specify standard linear SEMs
- Specifying correlation
- Using the command language to specify standard linear SEMs
- Specifying generalized SEMs: Family and link
- Specifying generalized SEMs: Family and link, multinomial logistic regression
- Specifying generalized SEMs: Family and link, paths from response variables
- Specifying generalized SEMs: Multilevel mixed effects (2 levels)
- Specifying generalized SEMs: Multilevel mixed effects (3 levels)
- Specifying generalized SEMs: Multilevel mixed effects (4+ levels)
- Specifying generalized SEMs: Multilevel mixed effects with random intercepts
- Specifying generalized SEMs: Multilevel mixed effects with random slopes
- Specifying generalized SEMs: Latent class analysis (LCA)
- Specifying generalized SEMs: Latent class analysis, class predictors
- Specifying generalized SEMs: Latent class analysis, two latent variables
Using path diagrams to specify standard linear SEMs

In structural equation modeling, models are often illustrated in a path diagram such as this one:

This diagram is composed of the following:

1. Boxes and circles with variable names written inside them.
   a. Boxes contain variables that are observed in the data.
   b. Circles contain variables that are unobserved, known as latent variables.

2. Arrows, called paths, that connect some of the boxes and circles.
   a. When a path points from one variable to another, it means that the first variable affects the second.
   b. More precisely, if $s \rightarrow d$, it means to add $\beta_k s$ to the linear equation for $d$. $\beta_k$ is called the path coefficient.
   c. Sometimes small numbers are written along the arrow connecting two variables. This means that $\beta_k$ is constrained to be the value specified. (Some authors use the term “path coefficient” to mean standardized path coefficient. We do not.)
   d. When no number is written along the arrow, the corresponding coefficient is to be estimated from the data. Sometimes symbols are written along the path arrow to emphasize this and sometimes not.
   e. The same path diagram used to describe the model can be used to display the results of estimation. In that case, estimated coefficients appear along the paths.

3. There are other elements that may appear on the diagram to indicate variances and between-variable correlations. We will get to them later.

Thus the above figure corresponds to the equations

\[
x_1 = \alpha_1 + \beta_1 X + e.x_1 \\
x_2 = \alpha_2 + \beta_2 X + e.x_2 \\
x_3 = \alpha_3 + \beta_3 X + e.x_3 \\
x_4 = \alpha_4 + \beta_4 X + e.x_4
\]
There's a third way of writing this model, namely,
\[(x1<-X) \ (x2<-X) \ (x3<-X) \ (x4<-X)\]
This is the way we would write the model if we wanted to use sem’s or gsem’s command syntax rather than drawing the model in the Builder. The full command we would type is
\[. \ sem \ (x1<-X) \ (x2<-X) \ (x3<-X) \ (x4<-X)\]
We will get to that later.

By the way, the above model is a linear single-level model. Linear single-level models can be fit by sem or by gsem. sem is the preferred way to fit linear single-level models because it has added features for these models that you might find useful later. Nonetheless, if you want to fit the model with gsem, you would type
\[. \ gsem \ (x1<-X) \ (x2<-X) \ (x3<-X) \ (x4<-X)\]
Whether we use sem or gsem, we obtain the same results.

However we write this model, what is it? It is a measurement model, a term loaded with meaning for some researchers. $X$ might be mathematical ability. $x_1$, $x_2$, $x_3$, and $x_4$ might be scores from tests designed to measure mathematical ability. $x_1$ might be the score based on your answers to a series of questions after reading this section.

The model we have just drawn, or written in mathematical notation, or written in Stata command notation, can be interpreted in other ways, too. Look at this diagram:

Despite appearances, this diagram is identical to the previous diagram except that we have renamed $x_4$ to be $y$. The fact that we changed a name obviously does not matter substantively. The fact that we have rearranged the boxes in the diagram is irrelevant, too; paths connect the same variables in the same directions. The equations for the above diagrams are the same as the previous equations with the substitution of $y$ for $x_4$:

\[
x_1 = \alpha_1 + X\beta_1 + e.x_1
\]
\[
x_2 = \alpha_2 + X\beta_2 + e.x_2
\]
\[
x_3 = \alpha_3 + X\beta_3 + e.x_3
\]
\[
y = \alpha_4 + X\beta_4 + e.y
\]

The Stata command notation changes similarly:
\[(x1<-X) \ (x2<-X) \ (x3<-X) \ (y<-X)\]
Many people looking at the model written this way might decide that it is not a measurement model but a measurement-error model. \( y \) depends on \( X \) but we do not observe \( X \). We do observe \( x_1, x_2, \) and \( x_3 \), each a measurement of \( X \), but with error. Our interest is in knowing \( \beta_4 \), the effect of true \( X \) on \( y \).

A few others might disagree and instead see a model for interrater agreement. Obviously, we have three or four raters who each make a judgment, and we want to know how well the judgment process works and how well each of these raters performs.

### Specifying correlation

One of the key features of SEM is the ease with which you can allow correlation between latent variables to adjust for the reality of the situation. In the measurement model in the previous section, we drew the following path diagram:

![Path Diagram](image)

which corresponded to the equations

\[
\begin{align*}
x_1 &= \alpha_1 + X\beta_1 + e.x_1 \\
x_2 &= \alpha_2 + X\beta_2 + e.x_2 \\
x_3 &= \alpha_3 + X\beta_3 + e.x_3 \\
x_4 &= \alpha_4 + X\beta_4 + e.x_4
\end{align*}
\]

\((X, x_1, x_2, x_3, x_4, e.x_1, e.x_2, e.x_3, e.x_4) \sim \text{i.i.d. with mean vector } \mu \text{ and covariance matrix } \Sigma\]

where i.i.d. means that observations are independent and identically distributed.

We must appreciate that \( \mu \) and \( \Sigma \) are estimated, just as are \( \alpha_1, \beta_1, \ldots, \alpha_4, \beta_4 \). Some of the elements of \( \Sigma \), however, are constrained to be 0; which elements are constrained is determined by how we specify the model. In the above diagram, we drew the model in such a way that we assumed that error variables were uncorrelated with each other. We could have drawn the diagram differently.
If we wish to allow for a correlation between $e.x_2$ and $e.x_3$, we add a curved path between the variables:

![Path Diagram]

The curved path states that there is a correlation to be estimated between the variables it connects. The absence of a curved path—say, between $e.x_1$ and $e.x_4$—means that the variables are constrained to be uncorrelated. That is not to say that $x_1$ and $x_4$ are uncorrelated; obviously, they are correlated because both are functions of the same $X$. Their corresponding error variables, however, are uncorrelated. The equations for this model, in their full detail, are

\[
\begin{align*}
x_1 &= \alpha_1 + X\beta_1 + e.x_1 \\
x_2 &= \alpha_2 + X\beta_2 + e.x_2 \\
x_3 &= \alpha_3 + X\beta_3 + e.x_3 \\
x_4 &= \alpha_4 + X\beta_4 + e.x_4
\end{align*}
\]

$$(X, x_1, x_2, x_3, x_4, e.x_1, e.x_2, e.x_3, e.x_4) \sim \text{i.i.d. with mean } \mu \text{ and variance } \Sigma$$
Σ is constrained such that

\[
\begin{align*}
\sigma_{e.x_1,e.x_2} &= \sigma_{e.x_2,e.x_1} = 0 \\
\sigma_{e.x_1,e.x_3} &= \sigma_{e.x_3,e.x_1} = 0 \\
\sigma_{e.x_1,e.x_4} &= \sigma_{e.x_4,e.x_1} = 0 \\
\sigma_{e.x_2,e.x_4} &= \sigma_{e.x_4,e.x_2} = 0 \\
\sigma_{e.x_3,e.x_4} &= \sigma_{e.x_4,e.x_3} = 0 \\
\sigma_{X,e.x_1} &= \sigma_{e.x_1,X} = 0 \\
\sigma_{X,e.x_2} &= \sigma_{e.x_2,X} = 0 \\
\sigma_{X,e.x_3} &= \sigma_{e.x_3,X} = 0 \\
\sigma_{X,e.x_4} &= \sigma_{e.x_4,X} = 0
\end{align*}
\]

μ is constrained such that

\[
\begin{align*}
\mu_X &= 0 \\
\mu_{e.x_1} &= 0 \\
\mu_{e.x_2} &= 0 \\
\mu_{e.x_3} &= 0 \\
\mu_{e.x_4} &= 0
\end{align*}
\]

The parameters to be estimated are

\[
\alpha_1, \beta_1, \ldots, \alpha_4, \beta_4,
\]

\[
\begin{align*}
\mu \\
\Sigma
\end{align*}
\]

Look carefully at the above list. You will not find a line reading

\[
\sigma_{e.x_2,e.x_3} = \sigma_{e.x_3,e.x_2} = 0
\]

although you will find lines constraining the other covariances between error terms to be 0. The line is missing because we drew a curved path between \(e.x_2\) and \(e.x_3\).

There are lots of other curved arrows we could have drawn. By not drawing them, we are asserting that the corresponding covariance is 0.
Some authors would draw the above model as

![Path diagram](image.png)

A curved path from a variable to itself indicates a variance. All covariances could be shown, including those between latent variables and between observed exogenous variables.

When we draw diagrams, however, we will assume variance paths and omit drawing them, and we will similarly assume but omit drawing covariances between observed exogenous variables (there are no observed exogenous variables in this model). The Builder in `sem` mode has an option concerning the latter. Like everyone else, we will not assume correlations between latent variables unless they are shown.

In `sem`’s (gsem’s) command-language notation, curved paths between variables are indicated via an option:

\[(x1<-X) (x2<-X) (x3<-X) (x4<-X), \text{cov}(e.x2*e.x3)\]

**Using the command language to specify standard linear SEMs**

You can describe your model to `sem` by using path diagrams with the Builder, or you can describe your model by using `sem`’s command language. Here are the trade-offs:

1. If you use path diagrams, you can see the results of your estimation as path diagrams or as standard computer output.
2. If you use the command language, only standard computer output is available.
3. Typing models in the command language is usually quicker than drawing them in the Builder.
4. You can type models in the command language and store them in do-files. By doing so, you can more easily correct the errors you make.
Translating from path diagrams to command language is easy.

1. Path diagrams have squares and circles to distinguish observed from latent variables. In the command language, variables are assumed to be observed if they are typed in lowercase and are assumed to be latent if the first letter is capitalized. Variable `educ` is observed, while variable `Knowledge` or `KNOWLEDGE` is latent.

If the observed variables in your dataset have uppercase names, type `rename _all, lower` to covert them to lowercase; see [D] `rename group`.

2. When typing path diagrams in the command language, remember the /// continuation line indicator. You may type

```
.semi (x1<-X) (x2<-X) (x3<-X) (y<-X)
```

or you may type

```
.semi (x1<-X) (x2<-X) ///
(x3<-X) (y<-X)
```

or you may type

```
.semi (x1<-X) ///
(x2<-X) ///
(x3<-X) ///
(y<-X)
```

3. In path diagrams, you draw arrows connecting variables to indicate paths. In the command language, you type variable names and arrows. So long as your arrow points the correct way, it does not matter which variable comes first. The following mean the same thing:

```
(x1 <- X)
(X -> x1)
```

4. In the command language, you may type multiple variables on either side of the arrow:

```
(X -> x1 x2 x3 x4)
```

The above means the same as

```
(X -> x1) (X -> x2) (X -> x3) (X -> x4)
```

which means the same as

```
(x1 <- X) (x2 <- X) (x3 <- X) (x4 <- X)
```

which means the same as

```
(x1 x2 x3 x4 <- X)
```

In a more complicated measurement model, we might have

```
(X Y -> x1 x2 x3) (X -> x4 x5) (Y -> x6 x7)
```

The above means the same as

```
(X -> x1 x2 x3 x4 x5) ///
(Y -> x1 x2 x3 x6 x7)
```
5. In path diagrams, you are required to show the error variables. In the command language, you may omit the error variables. `sem` knows that each endogenous variable needs an error variable. You can type

\[(x1 \leftarrow X) (x2 \leftarrow X) (x3 \leftarrow X) (x4 \leftarrow X)\]

and that means the same thing as

\[(x1 \leftarrow X e.x1) ///\]
\[(x2 \leftarrow X e.x2) ///\]
\[(x3 \leftarrow X e.x3) ///\]
\[(x4 \leftarrow X e.x4)\]

except that we have lost the small numeric 1s we had next to the arrows from \(e.x1\) to \(x1\), \(e.x2\) to \(x2\), and so on. To constrain the path coefficient, you type

\[(x1 \leftarrow X e.x1@1) ///\]
\[(x2 \leftarrow X e.x2@1) ///\]
\[(x3 \leftarrow X e.x3@1) ///\]
\[(x4 \leftarrow X e.x4@1)\]

It is easier to simply type

\[(x1 \leftarrow X) (x2 \leftarrow X) (x3 \leftarrow X) (x4 \leftarrow X)\]

but now you know that if you wanted to constrain the path coefficient \(x2 \leftarrow X\) to be 2, you could type

\[(x1 \leftarrow X) (x2 \leftarrow X@2) (x3 \leftarrow X) (x4 \leftarrow X)\]

If you wanted to constrain the path coefficients \(x2 \leftarrow X\) and \(x3 \leftarrow X\) to be equal, you could type

\[(x1 \leftarrow X) (x2 \leftarrow X@b) (x3 \leftarrow X@b) (x4 \leftarrow X)\]

We have not covered symbolic constraints with path diagrams yet, but typing symbolic names along the paths in either the Builder or the command language is how you constrain coefficients to be equal.

6. Curved paths are specified with the `cov()` option after you have specified your model:

\[(x1 x2 x3 x4 \leftarrow X), \text{cov}(e.x2*e.x3)\]

If you wanted to allow for correlation of \(e.x2*e.x3\) and \(e.x3*e.x4\), you can specify that in a single `cov()` option,

\[(x1 x2 x3 x4 \leftarrow X), \text{cov}(e.x2*e.x3 e.x3*e.x4)\]

or in separate `cov()` options:

\[(x1 x2 x3 x4 \leftarrow X), \text{cov}(e.x2*e.x3) \text{cov}(e.x3*e.x4)\]
7. Nearly all the above applies equally to \texttt{gsem}. We have to say “nearly” because sometimes, in some models, some concepts simply vanish. For instance, in a logistic model, there are no error terms. For generalized responses with family Gaussian, link log, there are error terms, but they cannot be correlated. Also, for responses with family Gaussian, link identity, and censoring, there are error terms, but they cannot be correlated. \texttt{gsem} also takes observed exogenous variables as given and so cannot estimate the covariances between them.

### Specifying generalized SEMs: Family and link

We began this discussion by showing you a linear measurement model:

\[
\begin{align*}
X & \quad x_1 \quad e.x_1 \\
& \quad x_2 \quad e.x_2 \\
& \quad x_3 \quad e.x_3 \\
& \quad x_4 \quad e.x_4
\end{align*}
\]

In this model, we observe continuous measurements \(x_1, \ldots, x_4\). What if \(x_1, \ldots, x_4\) were instead binary outcomes such as success and failure or passed and failed? That is, what if \(x_1, \ldots, x_4\) took on values of only 1 and 0?

In that case, we would want to fit a model appropriate to binary outcomes. Perhaps we want to fit a logistic regression model or a probit model. To do either one, we will have to use \texttt{gsem} rather than \texttt{sem}. We will use a probit model.

The path diagram for the measurement model with binary outcomes is

\[
\begin{align*}
X & \quad x_1 \quad \text{Bernoulli, probit} \\
& \quad x_2 \quad \text{Bernoulli, probit} \\
& \quad x_3 \quad \text{Bernoulli, probit} \\
& \quad x_4 \quad \text{Bernoulli, probit}
\end{align*}
\]

What were plain boxes for \(x_1, \ldots, x_4\) now say “Bernoulli” and “probit” at the top and bottom, meaning that the variable is from the Bernoulli family and is using the probit link. In addition, \(e.x_1, \ldots, e.x_4\) (the error terms for \(x_1, \ldots, x_4\)) have disappeared. In the generalized linear framework, there are error terms only for the Gaussian family.
Perhaps some math will clear up the issue. The generalized linear model is

$$g\{E(y \mid X)\} = \mathbf{x} \beta$$

and in the case of probit, $$g\{E(y \mid X)\} = \Phi^{-1}\{E(y \mid X)\}$$, where $$\Phi(\cdot)$$ is the cumulative normal distribution. Thus the equations are

$$\Phi^{-1}\{E(x_1 \mid X)\} = \alpha_1 + X \beta_1$$
$$\Phi^{-1}\{E(x_2 \mid X)\} = \alpha_2 + X \beta_2$$
$$\Phi^{-1}\{E(x_3 \mid X)\} = \alpha_3 + X \beta_3$$
$$\Phi^{-1}\{E(x_4 \mid X)\} = \alpha_4 + X \beta_4$$

Note that there are no error terms. Equivalently, the above can be written as

$$\Pr(x_1 = 1 \mid X) = \Phi(\alpha_1 + X \beta_1)$$
$$\Pr(x_2 = 1 \mid X) = \Phi(\alpha_2 + X \beta_2)$$
$$\Pr(x_3 = 1 \mid X) = \Phi(\alpha_3 + X \beta_3)$$
$$\Pr(x_4 = 1 \mid X) = \Phi(\alpha_4 + X \beta_4)$$

In gsem's command language, we write this model as

(x1 x2 x3 x4<-X, family(bernoulli) link(probit))

or as

(x1<-X, family(bernoulli) link(probit))
(x2<-X, family(bernoulli) link(probit))
(x3<-X, family(bernoulli) link(probit))
(x4<-X, family(bernoulli) link(probit))

In the command language, you can simply type probit to mean family(bernoulli) link(probit), so the model could also be typed as

(x1 x2 x3 x4<-X, probit)

or even as

(x1<-X, probit) (x2<-X, probit) (x3<-X, probit) (x4<-X, probit)

Whether you type family(bernoulli) link(probit) or type probit, when all the response variables are probit, you can type

(x1 x2 x3 x4<-X), probit

or

(x1<-X) (x2<-X) (x3<-X) (x4<-X), probit
The response variables do not have to be all from the same family and link. Perhaps $x_1$, $x_2$, and $x_3$ are pass/fail variables but $x_4$ is a continuous variable. Then the model would be diagrammed as

![Diagram showing the relationship between $X$, $x_1$, $x_2$, $x_3$, and $x_4$.]

The words “Gaussian” and “identity” now appear for variable $x_4$ and $e.x_4$ is back! Just as previously, the generalized linear model is

$$g\{E(y \mid X)\} = x\beta$$

and in the case of linear regression, $g(\mu) = \mu$, so our fourth equation becomes

$$E(x_4 \mid X) = X\beta_4$$

or

$$x_4 = \alpha + X\beta_4 + e.x_4$$

and the entire set of equations to be estimated is

$$\Pr(x_1 = 1 \mid X) = \Phi(\alpha_1 + X\beta_1)$$
$$\Pr(x_2 = 1 \mid X) = \Phi(\alpha_2 + X\beta_2)$$
$$\Pr(x_3 = 1 \mid X) = \Phi(\alpha_3 + X\beta_3)$$

$$x_4 = \alpha_4 + X\beta_4 + e.x_4$$

This can be written in the command language as

$$(x_1 \ x_2 \ x_3<-X, \text{family(bernoulli)} \ \text{link(probit))}$$
$$(x_4 \ <-X, \text{family(gaussian)} \ \text{link(identity))}$$

or as

$$(x_1 \ x_2 \ x_3<-X, \text{probit}) \ (x_4<-X, \text{regress})$$

`regress` is a synonym for `family(gaussian) link(identity)`. Because `family(gaussian) link(identity)` is the default, we can omit the option altogether:

$$(x_1 \ x_2 \ x_3<-X, \text{probit}) \ (x_4<-X)$$
We demonstrated generalized linear models above by using probit (family Bernoulli, link probit), but we could just as well have used logit (family Bernoulli, link logit). Nothing changes except that in the path diagrams, where you see probit, logit would now appear. Likewise, in the command, where you see `probit`, `logit` would appear.

The same is true with almost all the other possible generalized linear models. What they all have in common is

1. There are no error terms, except for family Gaussian.
2. Response variables appear the ordinary way except that the family is listed at the top of the box and the link is listed at the bottom of the box, except for family multinomial, link logit (also known as multinomial logistic regression or `mlogit`).

Concerning item 1, we just showed you a combined probit and linear regression with its $e.x_A$ term. Linear regression is family Gaussian.

Concerning item 2, multinomial logistic regression is different enough that we need to show it to you.

**Specifying generalized SEMs: Family and link, multinomial logistic regression**

Let’s consider a multinomial logistic model in which $y$ takes on one of four possible outcomes and is determined by $x_1$ and $x_2$. Such a model could be fit by Stata’s `mlogit` command:

```
. mlogit y x1 x2
```

The model could also be fit by `gsem`. The path diagram would be

![Path diagram for multinomial logistic regression](image)

Note that there are three boxes for $y$, boxes containing 2.$y$, 3.$y$, and 4.$y$. When specifying a multinomial logistic regression model in which the dependent variables can take one of $k$ different values, you draw $k - 1$ boxes. Names like 1.$y$, 2.$y$, ..., mean $y = 1$, $y = 2$, and so on. Logically speaking, you have to omit one of them. Which you omit is irrelevant and is known as the base outcome. Estimated coefficients in the model relevant to the other outcomes will be measured as a difference from the base outcome. We said which you choose is irrelevant, but it might be easier for you to interpret your model if you chose a meaningful base outcome, such as the most frequent outcome. In fact, that is just what Stata’s `mlogit` command does by default when you do not specify otherwise.
In path diagrams, you may implicitly or explicitly specify the base outcome. We implicitly specified the base by omitting 1.y from the diagram. We could have included it by drawing a box for 1.y and labeling it 1b.y. Stata understands the b to mean base category. See \[SEM\] Example 37g for an example.

Once you have handled specification of the base category, you draw path arrows from the predictors to the remaining outcome boxes. We drew paths from \(x_1\) and \(x_2\) to all the outcome boxes, but if we wanted to omit \(x_1\) to 3.y and 4.y, we could have omitted those paths.

Our example is simple in that \(y\) is the final outcome. If we had a more complex model where \(y\)'s outcome affected another response variable, arrows would connect all or some of 2.y, 3.y, and 4.y to the other response variable.

The command syntax for our simple example is

\[
(2.y 3.y 4.y<-x1 x2), mlogit
\]

2.y, 3.y, and 4.y are examples of Stata’s factor-variable syntax. The factor-variable syntax has some other features that can save typing in command syntax. i.y, for instance, means 1b.y, 2.y, 3.y, and 4.y. It is especially useful because if we had more levels, say, 10, it would mean 1b.y, 2.y, ..., 10.y. To fit the model we diagrammed, we could type

\[
(i.y<-x1 x2), mlogit
\]

If we wanted to include some paths and not others, we need to use the more verbose syntax. For instance, to omit paths from \(x_1\) to 3.y and 4.y, we would type

\[
(2.y<-x1 x2) (3.y 4.y<-x2), mlogit
\]

or

\[
(i.y<-x2) (2.y<-x1), mlogit
\]

For more information on specifying mlogit paths, see \[SEM\] Intro 3, \[SEM\] Example 37g, and \[SEM\] Example 41g.

**Specifying generalized SEMs: Family and link, paths from response variables**

When we draw a path from one response variable to another, we are stating that the first endogenous variable is a predictor of the other endogenous variable. The diagram might look like this:

![Path Diagram](image)

The command syntax might look like \((y1<-x1) (y2<-y1 x2)\).
The response variables in the model are linear, and note that there is a path from $y_1$ to $y_2$. Could we change $y_1$ to be a member of the generalized linear family, such as probit, logit, and so on? It turns out that we can:

$$x_1 \rightarrow y_1 \rightarrow y_2$$

In the command syntax, we could write this model as $(y_1<-x_1, \text{probit}) (y_2<-y_1 x_2)$. (In this case, the observed values and not the expectations are used to fit the $y_1->y_2$ coefficient. In general, this is true for all generalized responses that are not family Gaussian, link identity.)

We can make the substitution from linear to generalized linear if the path from $y_1$ appears in a recursive part of the model. We will define recursive shortly, but trust us that the above model is recursive all the way through. The substitution would not have been okay if the path had been in a nonrecursive portion of the model. The following is a through-and-through nonrecursive model:

$$x_1 \rightarrow y_1 \rightarrow y_2$$

It could be written in command syntax as $(y_1<-y_1 x_1 x_2) (y_2<-y_2 x_2 x_3)$.

In this model, we could not change $y_1$ to be family Bernoulli and link probit or any other generalized linear response variable. If we tried to fit the model with such a change, we would get an error message:

```
invalid path specification;
a loop among the paths between 'y1' and 'y2' is not allowed
r(198);
```

The software will spot the problem, but you can spot it for yourself.
Nonrecursive models have loops. Do you see the loop in the above model? You will if you work out the total effect of a change in \( y_2 \) from a change in \( y_1 \). Assume a change in \( y_1 \). Then that change directly affects \( y_2 \), the new value of \( y_2 \) affects \( y_1 \), which in turn indirectly affects \( y_2 \) again, which affects \( y_1 \), and on and on.

Now follow the same logic with either the probit or the continuous recursive models above. The change in \( y_1 \) affects \( y_2 \) and it stops right there.

We sympathize if you think that we have interchanged the terms recursive and nonrecursive. Remember it this way: total effects in recursive models can be calculated nonrecursively because the model itself is recursive, and total effects in nonrecursive models must be calculated recursively because the model itself is nonrecursive.

Anyway, you may draw paths from generalized linear response variables to other response variables, whether linear or generalized linear, as long as no loops are formed.

We gave special mention to multinomial logistic regression in the previous section because those models look different from the other generalized linear models. Multinomial logistic regression has a plethora of response variables. In the case of multinomial logistic regression, a recursive model with a path from the multinomial logistic outcome \( y_1 \) to (linear) \( y_2 \) would look like this:

In the command syntax, the model could be written as

\[
(2.y1 3.y1 4.y1<-x1, mlogit) (y2<-2.y1 3.y1 4.y1 x2)
\]

In multinomial logistic regression models, outcomes are numbered 1, 2, \ldots, \( k \). The outcomes might correspond to walk, take public transportation, drive a car, and fly. In ordered probit and logistic models, outcomes are also numbered 1, 2, \ldots, \( k \), and the outcomes are otherwise similar to multinomial logistic models except that the outcomes are also naturally ordered. The outcomes might be did poorly, did okay, did fairly well, and did well. After taking account of the ordering, you would probably want to model your remaining response variables in the same way you did for multinomial logistic regression. In that case, you would diagram the model like this:
In the command syntax, the model could be written as

\[(y_1<-x_1, \text{ologit}) \ (y_2<-2.y_1\ 3.y_1\ 4.y_1\ x_2)\]

Unlike multinomial logistic regression, in which the \(k\) outcomes result in the estimation of \(k - 1\) equations, in ordered probit and logistic models, only one equation is estimated, and thus the response variable is specified simply as \(y_1\) rather than \(2.y_1\), \(3.y_1\), and \(4.y_1\). Even so, you probably will want the different outcomes to have separate coefficients in the \(y_2\) equation so that the effects of being in groups 1, 2, 3, and 4 are \(\beta_0\), \(\beta_0 + \beta_1\), \(\beta_0 + \beta_2\), and \(\beta_0 + \beta_3\), respectively. If you drew the diagram with a single path from \(y_1\) to \(y_2\), the effects of being in the groups would be \(\beta_0 + 1\beta_1\), \(\beta_0 + 2\beta_1\), \(\beta_0 + 3\beta_1\), and \(\beta_0 + 4\beta_1\), respectively.

**Specifying generalized SEMs: Multilevel mixed effects (2 levels)**

The models above are all single-level models or, equivalently, observational-level models. Let’s reconsider our original measurement model:
The data for this model would look something like this:

```
. list in 1/5

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>96</td>
<td>82</td>
<td>96</td>
<td>43</td>
</tr>
<tr>
<td>2</td>
<td>106</td>
<td>81</td>
<td>93</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>127</td>
<td>95</td>
<td>96</td>
<td>134</td>
</tr>
<tr>
<td>4</td>
<td>123</td>
<td>98</td>
<td>94</td>
<td>109</td>
</tr>
<tr>
<td>5</td>
<td>119</td>
<td>99</td>
<td>100</td>
<td>108</td>
</tr>
</tbody>
</table>
```

Let’s pretend that the observations are students and that \(x_1, \ldots, x_4\) are four test scores.

Now let’s pretend that we have new data with students from different schools. A part of our data might look like this:

```
. list in 1/10

<table>
<thead>
<tr>
<th>school</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>116</td>
<td>128</td>
<td>125</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>92</td>
<td>104</td>
<td>101</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>105</td>
<td>117</td>
<td>114</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>117</td>
<td>129</td>
<td>126</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>102</td>
<td>114</td>
<td>111</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>116</td>
<td>122</td>
<td>120</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>95</td>
<td>101</td>
<td>99</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>80</td>
<td>86</td>
<td>84</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>96</td>
<td>102</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>107</td>
<td>113</td>
<td>111</td>
</tr>
</tbody>
</table>
```

We have four test scores from various students in various schools. Our corresponding model might be

The `school1` inside double circles in the figure tells us, “I am a latent variable at the school level—meaning that I am constant within school and vary across schools—and I correspond to the latent variable named \(M_1\).” That is, double circles denote a latent variable named \(M#\), where \# is the subscript of the variable name inside the double circles. Meanwhile, the variable name inside the double circles specifies the level of the \(M#\) variable.
The equivalent command-language form for this model is

\[(x_1 \ x_2 \ x_3 \ x_4 \leftarrow X \ M1[\text{school}])\]

The \(M1\) part of \(M1[\text{school}]\) is the latent variable’s name, while the \([\text{school}]\) part means “at the school level”. Thus \(M1[\text{school}]\) denotes latent variable \(M1\), which varies at the school level or, equivalently, which is constant within school.

By the way, we gave the latent variable the name \(M1\) just to match what the Builder does by default. In real life, we would probably give it a different name, such as \(S\).

The mathematical way of writing this model is

\[
x_1 = \alpha_1 + \beta_1 X + \gamma_1 M_{1,S} + e.x_1 \\
x_2 = \alpha_2 + \beta_2 X + \gamma_2 M_{1,S} + e.x_2 \\
x_3 = \alpha_3 + \beta_3 X + \gamma_3 M_{1,S} + e.x_3 \\
x_4 = \alpha_4 + \beta_4 X + \gamma_4 M_{1,S} + e.x_4
\]

where \(S = \text{school number}\).

Thus we have three different ways of referring to the same thing: \(\text{school1}\) inside double circles in the path diagram corresponds to \(M1[\text{school}]\) in the command language, which in turn corresponds to \(M_{1,S}\) in the mathematical notation.

Rabe-Hesketh, Skrondal, and Pickles (2004) use boxes to identify different levels of the model. Our path diagrams are similar to theirs, but they do not use double circles for multilevel latent variables. We can put a box around the individual-level part of the model, producing something that looks like this:

You can do that in the Builder, but the box has no special meaning to \textit{gsem}; however, adding the box does make the diagram easier to understand in presentations.

However you diagram the model, this model is known as a two-level model. The first level is the student or observational level, and the second level is the school level.
Specifying generalized SEMs: Multilevel mixed effects (3 levels)

A three-level model adds another latent variable at yet another level. For instance, perhaps we have data on four test scores for individual students, who are nested within school, which are nested within counties. The beginning of our data might look like this:

```
. list in 1/10
  county  school  x1  x2  x3  x4
1.        1       103 122 117 120
2.        1        88 107 102 105
3.        1       106 125 120 123
4.        1       108 127 122 125
5.        1       100 119 114 117
6.        2        90  99  97  98
7.        2        87  96  94  95
8.        2       121 130 128 129
9.        2       100 109 107 108
10.       2        88  97  95  96
```

Further down, the data might look like this:

```
. list in 991/1
  county  school  x1  x2  x3  x4
991.      2       100  96  97  97
992.      2        99  95  96  96
993.      2        90  86  87  87
994.      2        79  75  76  76
995.      2        68  64  65  65
996.      2       105 111 109 110
997.      2        94 100  98  99
998.      2        90  96  94  95
999.      2        99 105 103 104
1000.     2       104 110 108 109
```
We might diagram the student-within-school-within-county model as

![Diagram of the student-within-school-within-county model]

Now we have two variables in double circles: \texttt{county1} and \texttt{school2}.

\texttt{county1} tells us, “I am a latent variable at the \texttt{county} level—meaning that I am constant within county and vary across counties—and I correspond to the latent variable named \texttt{M1}.”

\texttt{school2} tells us, “I am a latent variable at the \texttt{school} level—meaning that I am constant within school and vary across schools—and I correspond to the latent variable named \texttt{M2}.”

Do not read anything into the fact that the latent variables are named \texttt{M1} and \texttt{M2} rather than \texttt{M2} and \texttt{M1}. When we diagrammed the figure in the Builder, we diagrammed \texttt{county} first and then we added \texttt{school}. Had we diagrammed them the other way around, the latent variable names would have been reversed. We can edit the names after the fact, but we seldom bother.

The equivalent command-language form for this three-level model is

\[(x1 \ x2 \ x3 \ x4<-X \ M1[county] \ M2[county>school])\]

or

\[(x1 \ x2 \ x3 \ x4<-X \ M1[county] \ M2[school<county])\]

The \texttt{[school<county]} part is usually spoken aloud as “school within county”, and \texttt{[county>school]} is spoken aloud as “county containing school”. When we write \texttt{[county>school]} or \texttt{[school<county]}, we are saying that the counties contain schools or, equivalently, that school is nested within county. Thus the model we are specifying is in fact a three-level nested model.

The order in which we specify the level effects is irrelevant; we could just as well type

\[(x1 \ x2 \ x3 \ x4<-X \ M2[county>school] \ M1[county])\]
or

\[(x_1 \ x_2 \ x_3 \ x_4 \leftarrow X \ M2[\text{school}<\text{county}] \ M1[\text{county}])\]

The mathematical way of writing this model is

\[
x_1 = \alpha_1 + \beta_1 X + \gamma_1 M_{1,C} + \delta_1 M_{2,S} + e.x_1
\]
\[
x_2 = \alpha_2 + \beta_2 X + \gamma_2 M_{1,C} + \delta_2 M_{2,S} + e.x_2
\]
\[
x_3 = \alpha_3 + \beta_3 X + \gamma_3 M_{1,C} + \delta_3 M_{2,S} + e.x_3
\]
\[
x_4 = \alpha_4 + \beta_4 X + \gamma_4 M_{1,C} + \delta_4 M_{2,S} + e.x_4
\]

where \(C\) = county number and \(S\) = school number.

Three-level nested models are often double-boxed (or double-shaded) in presentation:

The darker box highlights the first level (student, in our case), and the lighter box highlights the second level (school, in our case). As we previously mentioned, you can add these boxes using the Builder, but they are purely aesthetic and have no meaning to \textit{gsem}.

**Specifying generalized SEMs: Multilevel mixed effects (4+ levels)**

We have now diagrammed one-, two-, and three-level nested models. You can draw with the Builder and fit with \textit{gsem} higher-level models, but you will usually need lots of data to be successful in fitting those models and, even so, you may run into other estimation problems.
Specifying generalized SEMs: Multilevel mixed effects with random intercepts

Let’s change gears and consider with the following simple model a linear regression of $y$ on $x_1$ and $x_2$:

$$y = \alpha + \beta x_1 + \gamma x_2 + e.y$$

In the command language, this model is specified as

$$(x_1 \ x_2 \rightarrow y)$$

or

$$(y \leftarrow x_1 \ x_2)$$

and the mathematical representation is

$$y = \alpha + \beta x_1 + \gamma x_2 + e.y$$

Now assume that we have data not just on $y$, $x_1$, and $x_2$, but also on county of residence. If we wanted to add a random intercept—a random effect—for county, the diagram becomes

The command-language equivalent is

$$(x_1 \ x_2 \ M1[county] \rightarrow y)$$

or

$$(y \leftarrow x_1 \ x_2 \ M1[county])$$
and the mathematical representation is

\[ y = \alpha + \beta x_1 + \gamma x_2 + M_{1,C} + e.y \]

where \( C \) = county number. Actually, the model is

\[ y = \alpha + \beta x_1 + \gamma x_2 + \delta M_{1,C} + e.y \]

but \( \delta \) is automatically constrained to be 1 by \texttt{gsem}. The software is not reading our mind; consider a solution for \( M_{1,C} \) with \( \delta = 0.5 \). Then another equivalent solution is \( M_{1,C}/2 \) with \( \delta = 1 \), and another is \( M_{1,C}/4 \) with \( \delta = 2 \), and on and on, because in all cases, \( \delta M_{1,C} \) will equal the same value. The fact is that \( \delta \) is unidentified. Whenever a latent variable is unidentified because of such scaling considerations, \texttt{gsem} (and \texttt{sem}) automatically set the coefficient to 1 and thus set the scale.

This is a two-level model: the first level is the observational level and the second level is county.

Just as we demonstrated previously with the measurement model, we could have a three-level nested model. We could imagine the observational level nested within the county level nested within the state level. The path diagram would be

![Path Diagram](null)

The command-language equivalent is

\[(y<-x1 \ x2 \ M1[county<state] \ M2[state])\]

and the mathematical representation is

\[ y = \alpha + \beta x_1 + \gamma x_2 + M_{1,C} + M_{2,S} + e.y \]

where \( C \) = county number and \( S \) = state number. Just as previously, the actual form of the equation is

\[ y = \alpha + \beta x_1 + \gamma x_2 + \delta M_{1,C} + \zeta M_{2,S} + e.y \]

but the constraints \( \delta = \zeta = 1 \) are automatically applied by the software.
You can specify higher-level models, but just as we mentioned when discussing the higher-level measurement models, you will need lots of data to fit them successfully and you still may run into other estimation problems.

You can also fit crossed models, such as county and occupation. Unlike county and state, where a particular county appears only in one state, the same occupation will appear in more than one county, which is to say, occupation and county are crossed, not nested. Except for a change in the names of the variables, the path diagram for this model looks identical to the diagram for the state and county-within-state model:

When we enter the model into the Builder, however, we will specify that the two effects are crossed rather than nested.

The command-language way of expressing this crossed model is

\[(y<-x1 \ x2 \ M1[county] \ M2[occupation])\]

and the mathematical representation is

\[y = \alpha + \beta x_1 + \gamma x_2 + M_{1,C} + M_{2,O} + e.y\]

where \(C = \text{county number}\) and \(O = \text{occupation number}\).

Higher-order crossed models are theoretically possible, and you will find \texttt{gsem} game to try to fit them. However, you are unlikely to be successful unless you have lots of data so that all the effects can be identified. In fact, you can specify crossed models anywhere you can specify nested models, but the same comment applies.
Specifying generalized SEMs: Multilevel mixed effects with random slopes

Perhaps we feel that county does not affect the intercept, but it affects the slope of $x_1$. In that case, we just shift the path from `county1` to instead point to the path between $y$ and $x_1$.

The command-language equivalent for this model is

\[(y<-x_1 \text{.} \text{c.x1#M1[county]}) (y<-x_2)\]

or

\[(y<-x_1 \text{.} \text{c.x1#M1[county]} x_2)\]

To include a random slope (coefficient) on a variable in the command language, include the variable on which you want the random slope just as you ordinarily would:

\[(y<-x_1)\]

Then, before closing the parenthesis, type

\[c.\text{variable}\#\text{latent_variable}[\text{grouping_variable}]\]

In our case, the variable on which we want the random coefficient is $x_1$, the latent variable we want to create is named M1, and the grouping variable within which the latent variable will be constant is county, so we type

\[(y<-x_1 \text{.} \text{c.x1#M1[county]})\]

Finally, finish off the command by typing the rest of the model:

\[(y<-x_1 \text{.} \text{c.x1#M1[county]} x_2)\]

This is another example of Stata’s factor-variable notation.

The mathematical representation of the random-slope model is

\[y = \alpha + \beta x_1 + \gamma x_2 + \delta M_1, C x_1 + e.y\]

where $C$ = county number and $\delta = 1$. 
You can simultaneously specify both random slope and random intercept by putting together what we have already done. The first time you see the combined path diagram, you may think there is a mistake:

County appears twice in two different double circles, once with a subscript of 1 and the other with a subscript of 2! We would gamble at favorable odds that you would have included county in double circles once and then would have drawn two paths from it, one to the path between x1 and y, and the other to y itself. That model, however, would make an extreme assumption.

Consider what you would be saying. There is one latent variable M1. The random intercepts would be equal to M1. The random slopes would be equal to γM1, a scalar replica of the random intercepts! You would be constraining the random slopes and intercepts to be correlated 1.

What you usually want, however, is one latent variable for the random slopes and another for the random intercepts. They might be correlated, but they are not related by a multiplicative constant. Thus we also included a covariance between the two random effects.

The command-language equivalent of the correct path diagram (the one shown above) is

\[(y<-x1 \, x2 \, c.x1#M2[county] \, M1[county])\]

You will learn later that the command language assumes covariances exist between latent exogenous variables unless an option is specified. Meanwhile, the Builder assumes those same covariances are 0 unless a curved covariance path is drawn.

The mathematical representation of the model is

\[y = \alpha + \beta x_1 + \gamma x_2 + \delta M_{2,C} x_1 + \zeta M_{1,C} + e.y\]

where \(C = \text{county number}\) and \(\delta = \zeta = 1\).

**Specifying generalized SEMs: Latent class analysis (LCA)**

All the latent variables discussed in the previous sections are continuous latent variables. We can also fit models with categorical latent variables using \texttt{gsem}. The unobserved levels of a categorical latent variable are known as latent classes. The classes represent groups in a population such as groups of consumers with different buying preferences. In this manual, we refer to models with categorical latent variables as latent class models, and we refer to an analysis involving categorical latent variables as a latent class analysis (LCA).
Latent class models have distinct parameters for each class. We also estimate the probability of being in each class.

Let’s consider a simple latent class model with a categorical latent variable $C$ that has two classes and with three observed binary variables, $x_1$, $x_2$, and $x_3$, that are modeled using logistic regression. Some textbooks draw a path diagram for this model as a measurement model. For instance, you may see something like

When we fit this model, we will actually estimate one intercept for $x_1$ in class 1 and one intercept for $x_1$ in class 2. Likewise, we will estimate class-specific intercepts for $x_2$ and for $x_3$. Mathematically, the logistic equations for class 1 are

\[
\Pr(x_1 = 1 | C = 1) = \frac{\exp(\alpha_{11})}{1 + \exp(\alpha_{11})}
\]

\[
\Pr(x_2 = 1 | C = 1) = \frac{\exp(\alpha_{21})}{1 + \exp(\alpha_{21})}
\]

\[
\Pr(x_3 = 1 | C = 1) = \frac{\exp(\alpha_{31})}{1 + \exp(\alpha_{31})}
\]

and the equations for class 2 are

\[
\Pr(x_1 = 1 | C = 2) = \frac{\exp(\alpha_{12})}{1 + \exp(\alpha_{12})}
\]

\[
\Pr(x_2 = 1 | C = 2) = \frac{\exp(\alpha_{22})}{1 + \exp(\alpha_{22})}
\]

\[
\Pr(x_3 = 1 | C = 2) = \frac{\exp(\alpha_{32})}{1 + \exp(\alpha_{32})}
\]

In a sense, $C$ has an effect on $x_1$, $x_2$, and $x_3$, but we do not estimate any coefficients on $C$. Rather, the two classes of $C$ partition our model into two component models, each with the same $x$ variables but with different coefficients (intercepts) on those $x$ variables. In Stata’s path diagrams, arrows always represent coefficients, so the path diagram above does not actually correspond with the model we fit. A more accurate description would be to draw the diagram as
for the first class and again as

for the second class. Now, we could place the six estimated intercepts in the six boxes.

However, these two diagrams do not fully describe the model. We also estimate the probability of being in each class using multinomial logistic regression,

\[
\begin{align*}
\Pr(C = 1) &= \frac{e^{\gamma_1}}{e^{\gamma_1} + e^{\gamma_2}} \\
\Pr(C = 2) &= \frac{e^{\gamma_2}}{e^{\gamma_1} + e^{\gamma_2}}
\end{align*}
\]

where \( \gamma_1 \) and \( \gamma_2 \) are intercepts in the multinomial logit model. By default, the first class will be treated as the base, so \( \gamma_1 = 0 \). We do not have a place for the \( \gamma \) parameter estimates on our diagrams above. Thus, we cannot draw path diagrams in the Builder to fit latent class models. Instead, we fit latent class models using the command language.

We specify this model using the command language as

\[(x_1 \ x_2 \ x_3 \ <- \_), \logit \ lclass(C \ 2)\]

\( lclass(C \ 2) \) says that our categorical latent variable is named \( C \) and has two classes. Because we are only estimating an intercept in each equation, we do not include any variables on the right side of the arrow. We could directly specify that a constant, _cons, is included by typing

\[(x_1 \ x_2 \ x_3 \ <- \ _cons) \ \logit \ lclass(C \ 2)\]

We could also be more explicit about having separate models for the two classes by including 1: at the beginning of the path specification for class 1 and including 2: at the beginning of the path specification for class 2.

\[(1: \ x_1 \ x_2 \ x_3 \ <- \ _cons) \ (2: \ x_1 \ x_2 \ x_3 \ <- \ _cons), \logit \ lclass(C \ 2)\]

We could even write a separate path specification for each class and outcome variable combination.

\[(1: \ x_1 \ <- \ _cons) \ (1: \ x_2 \ <- \ _cons) \ (1: \ x_3 \ <- \ _cons) \ (2: \ x_1 \ <- \ _cons) \ (2: \ x_2 \ <- \ _cons) \ (2: \ x_3 \ <- \ _cons), \logit \ lclass(C \ 2)\]

In any case, the model is the same.
We can build on this to fit different models for different classes and for different outcome variables. For instance, if x3 is ordinal, we could use an ordinal logit model for this variable by typing

```
(1: x1 <- _cons, logit) (1: x2 <- _cons, logit)
(1: x3 <- _cons, ologit) (2: x1 <- _cons, logit)
(2: x2 <- _cons, logit) (2: x3 <- _cons, ologit),
lclass(C 2)
```

Or we can add a constraint that the intercepts for x1 and x2 are equal in class 2.

```
(1: x1 <- _cons) (1: x2 <- _cons) (1: x3 <- _cons)
(2: x1 <- _cons@c) (2: x2 <- _cons@c) (2: x3 <- _cons),
logit lclass(C 2)
```

Other than including continuous latent variables, all command-language features of `gsem` discussed above can be used to modify and extend this latent class model.

### Specifying generalized SEMs: Latent class analysis, class predictors

We can include variables in a latent class model that predict class membership. Extending the model above, we simply add exogenous variables to the multinomial logit model for C. If we believe that z predicts class membership, our multinomial logit model becomes

\[
\begin{align*}
\Pr(C = 1) &= \frac{e^{\gamma_1 + z\beta_1}}{e^{\gamma_1 + z\beta_1} + e^{\gamma_2 + z\beta_2}} \\
\Pr(C = 2) &= \frac{e^{\gamma_2 + z\beta_2}}{e^{\gamma_1 + z\beta_1} + e^{\gamma_2 + z\beta_2}}
\end{align*}
\]

where \(\gamma_1 = 0\) and \(\beta_1 = 0\) when \(C = 1\) is the base class.

To fit this model using the command language, we type

```
(x1 x2 x3 <- _cons, logit) (C <- z), lclass(C 2)
```

We can also allow different predictors for different classes. For instance, if C has three levels, we can type

```
(x1 x2 x3 <- _cons, logit)
(1.C <- z1 z2 z3)
(2.C <- z1 z2)
(3.C <- z1 z2 z3),
lclass(C 3)
```

where z1, z2, and z3 are predictors of class membership, but only z1 and z2 are predictors of \(C = 2\).

### Specifying generalized SEMs: Latent class analysis, two latent variables

Latent class models can include more than one categorical latent variable. Let’s consider an example where C has two classes and D has three classes.

We could extend our model from *Specifying generalized SEMs: Latent class analysis (LCA)* to include D by typing

```
(x1 x2 x3 <- _cons), logit lclass(C 2) lclass(D 3)
```
Now, if we want to refer to a specific class, we use factor-variable notation before the colon. The most verbose specification of this model is

\[
(1.\text{C#1.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#1.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#1.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#1.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#1.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#1.D}: x_3 \leftarrow \_\text{cons}) \\
(1.\text{C#2.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#2.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#2.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#2.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#2.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#2.D}: x_3 \leftarrow \_\text{cons}) \\
(1.\text{C#3.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#3.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#3.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#3.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#3.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#3.D}: x_3 \leftarrow \_\text{cons}), \\
\text{logit} \ l\text{class(C 2)} \ l\text{class(D 3)}
\]

As before, this allows us to specify different models for different outcomes or for different classes. It also allows us to specify constraints. For instance, if we want equal intercepts for \(x_3\) when \(C = 2\) and \(D = 2\) and when \(C = 2\) and \(D = 3\), we type

\[
(1.\text{C#1.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#1.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#1.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#1.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#1.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#1.D}: x_3 \leftarrow \_\text{cons}) \\
(1.\text{C#2.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#2.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#2.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#2.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#2.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#2.D}: x_3 \leftarrow \_\text{cons@a}) \\
(1.\text{C#3.D}: x_1 \leftarrow \_\text{cons}) (1.\text{C#3.D}: x_2 \leftarrow \_\text{cons}) (1.\text{C#3.D}: x_3 \leftarrow \_\text{cons}) \\
(2.\text{C#3.D}: x_1 \leftarrow \_\text{cons}) (2.\text{C#3.D}: x_2 \leftarrow \_\text{cons}) (2.\text{C#3.D}: x_3 \leftarrow \_\text{cons@a}), \\
\text{logit} \ l\text{class(C 2)} \ l\text{class(D 3)}
\]

We can allow predictors of class membership just as we did with a single categorical latent variable. For specifying \(z\) as a predictor of the classes of \(C\), we type

\[(x_1 \ x_2 \ x_3 \leftarrow \_\text{cons}, \text{logit}) \ (C \leftarrow z), \ l\text{class(C 2)} \ l\text{class(D 3)} \]

We can also add predictors of specific cells in the interaction between \(C\) and \(D\). For instance,

\[(x_1 \ x_2 \ x_3 \leftarrow \_\text{cons}, \text{logit}) \ (1.\text{C#3.D} \leftarrow z), \ l\text{class(C 2)} \ l\text{class(D 3)} \]

Adding predictors or even an intercept (by specifying \_\text{cons}) to an individual cell induces correlation between the categorical latent variables \(C\) and \(D\).

And now you, too, are an expert on Stata’s path diagrams and command language for SEM.

Reference


Also see

- [SEM] Intro 1 — Introduction
- [SEM] Intro 3 — Learning the language: Factor-variable notation (gsem only)
- [SEM] Builder — SEM Builder
- [SEM] Builder, generalized — SEM Builder for generalized models
- [SEM] sem and gsem path notation — Command syntax for path diagrams
This entry concerns \texttt{gsem} only; \texttt{sem} does not allow the use of factor variables. \texttt{gsem} allows you to use Stata’s factor-variable notation in path diagrams (in the SEM Builder) and in the command language. Use of the notation, though always optional, is sometimes useful:

1. Use of factor variables sometimes saves effort. For instance, rather than typing

\begin{verbatim}
. generate femXage = female * age
\end{verbatim}

and including \texttt{femXage} in your model, you can directly include \texttt{1.female#c.age}. You can type \texttt{1.female#c.age} with the \texttt{gsem} command or type it into an exogenous-variable box in a path diagram.

2. Use of factor variables can save even more effort in command syntax. You can type things like \texttt{i.female} \texttt{i.skill} \texttt{i.female#i.skill} to include main effects and interaction effects of female interacted with indicators for all the different levels of skill.

3. Use of factor variables causes the postestimation commands \texttt{margins}, \texttt{contrast}, and \texttt{pwcompare} to produce more useful output. For instance, they will show the effect of the discrete change between \texttt{female} = 0 and \texttt{female} = 1 rather than the infinitesimal change (slope) of \texttt{female}.

In the examples in the rest of this manual, we sometimes use factor-variable notation and, at other times, ignore it. We might have variable \texttt{smokes} recording whether a person smokes. In one example, you might see \texttt{smokes} directly included in the model and yet, in another example, we might have the odd-looking \texttt{1.smokes}, which is factor-variable notation for emphasizing that \texttt{smokes} is a 0/1 variable. We probably used \texttt{1.smokes} for reason 3, although we might have done it just to emphasize that variable \texttt{smokes} is indeed 0/1.

In other examples, we will use more complicated factor-variable notation, such as \texttt{1.female#c.age}, because it is more convenient.

You should follow the same rules. Use factor-variable notation when convenient or when you plan subsequently to use postestimation commands \texttt{margins}, \texttt{contrast}, or \texttt{pwcompare}. If neither reason applies, use factor-variable notation or not. There is no benefit from consistency in this case.

Remarks and examples

Remarks are presented under the following headings:

- Specifying indicator variables
- Specifying interactions with indicator variables
- Specifying categorical variables
- Specifying interactions with categorical variables
- Specifying endogenous variables
Specifying indicator variables

We wish to fit the following model:

\[ y_i = \beta_0 + \beta_1 x_i + \epsilon_i. \]

This model corresponds to the equation \( y_i = \beta x_i + \epsilon_i \). It can be fit using the Builder with the path diagram above or using the command syntax by typing

\[ \texttt{. gsem (y<-x)} \]

Say we now tell you that \( x \) is a 0/1 (binary) variable. Perhaps \( x \) indicates male/female. That changes nothing mathematically. The way we drew the path diagram and specified the command syntax are as valid when \( x \) is an indicator variable as they were when \( x \) was continuous.

Specifying \( 1.x \) is a way you can emphasize that \( x \) is 0/1:

\[ 1.x \rightarrow y \rightarrow \epsilon_i \]

That model will produce the same results as the first model. In command syntax, we can type

\[ \texttt{. gsem (y<-1.x)} \]

The only real advantage of \( 1.x \) over \( x \) arises when we plan to use the postestimation command \texttt{margins}, \texttt{contrast}, or \texttt{pwcompare}, as we mentioned above. If we fit the model using \( 1.x \), those commands will know \( x \) is 0/1 and will exploit that knowledge to produce more useful output.

In other cases, the \#.varname notation can sometimes save us effort. \#.varname means a variable equal to 1 if \texttt{varname} = \#. It is not required that varname itself be an indicator variable.

Let's say variable \texttt{skill} takes on the values 1, 2, and 3 meaning unskilled, skilled, and highly skilled. Then \( 3.skill \) is an indicator variable for \( \texttt{skill} = 3 \). We could use \( 3.skill \) in path diagrams or in the command language just as we previously used \( 1.x \).

Specifying interactions with indicator variables

We can specify interactions with \#. Say we wish to fit the model

\[ y_i = \beta_0 + \beta_1 \text{female}_i + \beta_2 \text{age}_i + \beta_3 \text{female}_i \times \text{age}_i + \epsilon_i \]
We could do that by specifying `1.female`, `age`, and `1.female#c.age` in a path diagram:

![Path diagram](image)

We can use `1.female`, `age`, and `1.female#c.age` in command syntax, too:

```
.gsem (y<-1.female age 1.female#c.age)
```

The `c` in `c.age` specifies that `age` is a continuous variable.

We can just as easily create interactions of indicator variables with other indicator variables. If variable `employed` is a 0/1 variable, then `1.female#1.employed` creates an indicator for employed female.

**Specifying categorical variables**

We use the same `.varname` notation to handle categorical variables as we did for indicator variables. Consider the following model:

![Path diagram](image)

The same model would be specified in command syntax as

```
.gsem (y<-1.female 2.skill 3.skill)
```

The equation for the model is

\[
y_i = \beta_0 + \beta_1 1.\text{female}_i + \beta_2 2.\text{skill}_i + \beta_3 3.\text{skill}_i + \epsilon_i
\]

However we express the model, we are including indicator variables for skill level 2 and for skill level 3. We intentionally omit skill level 1.

Categorical variables require a base level, a level against which the other effects will be measured. We omitted `1.skill` to cause `1.skill` to become the base level. This results in $\beta_2$ measuring the difference in $y$ between skill levels 2 and 1, and $\beta_3$ measuring the difference in $y$ between skill levels 3 and 1.
Omission is one way to specify the base level. The other way is to include all the skill levels and indicate which one we want to be used as the base:

\[
\begin{align*}
1.\text{female} \\
1\text{b.skill} \\
2.\text{skill} \\
3.\text{skill} \\
y \\
\varepsilon_1
\end{align*}
\]

In command syntax, we type

```
. gsem (y<-1.\text{female} 1\text{b.skill} 2.\text{skill} 3.\text{skill})
```

b specifies the base level. Specifying 1b.skill makes skill level 1 the base. Had we wanted skill level 2 to be the base, we would have specified 2b.skill.

Specification by omission is usually easier for simple models. For more complicated models, which might have paths from different skill levels going to different variables, it is usually better to specify all the skill levels and all the relevant paths, mark one of the skill levels as the base, and let \texttt{gsem} figure out the correct reduced-form model.

By the way, indicator variables are just a special case of categorical variables. Indicator variables are categorical variables with two levels. Everything just said about categorical variable skill applies equally to indicator variable female. We can specify 1.\text{female} by itself, and thus implicitly specify that female \( \neq 1 \) is the base, or we can explicitly specify both 0b.\text{female} and 1.\text{female}.

The command language has a neat syntax for specifying all the indicator variables that can be manufactured from a categorical variable. When you type \texttt{i.skill}, it is the same as typing 1b.skill 2.skill 3.skill. You can use the \texttt{i.} shorthand in the command language by typing

```
. gsem (y<-1.\text{female} \text{i.skill})
```

or even by typing

```
. gsem (y<-i.\text{female} \text{i.skill})
```

Most experienced Stata command-line users would type the model the second way, putting \texttt{i.} in front of both \texttt{skill} and \texttt{female}. They would not bother to think whether variables are indicator or categorical, nor would they concern themselves with remembering how the indicator and categorical variables are coded.

You cannot use the \texttt{i.} notation in path diagrams, however. Path diagrams allow only one variable per box. The \texttt{i.} notation produces at least two variables, and it usually produces a lot more.
Specifying interactions with categorical variables

We just discussed the model \((y\leftarrow i.\text{female} \ i.\text{skill})\). Obviously, the next step is to include an interaction between female and skill level. To write such a model, we use the same \# interaction indicator that we have already discussed in the context of indicator variables: \((y\leftarrow i.\text{female} \ i.\text{skill} \ i.\text{female}\#i.\text{skill})\). We use the same notation because there is really no distinction between factor variables and indicator variables. We can specify the model exactly as shown by using the command syntax:

\[
\text{. gsem (y\leftarrow i.\text{female} \ i.\text{skill} \ i.\text{female}\#i.\text{skill})}
\]

In the Builder, we could diagram this model by specifying all the individual interactions but omitting the base levels,
or by including them,

```
. gsem (y<-1.female 2.skill 3.skill 1.female#2.skill 1.female#3.skill)
. gsem (y<-1.female 1b.skill 2.skill 3.skill 1.female#1b.skill 1.female#2.skill 1.female#3.skill)
```

But in command syntax, it is easiest to type

```
. gsem (y<-1.female i.skill i.female#i.skill)
```

### Specifying endogenous variables

You may not use factor variables to specify endogenous variables except for the multinomial logistic regression model. Multinomial logistic regression is also known as mlogit and as being family multinomial, link logit.

In the mlogit case, you must specify a base level, which you can do implicitly by omission or explicitly by including a `b` on one of the levels. See [SEM] Example 37g and [SEM] Example 41g for examples.
Also see

[SEM] Intro 2 — Learning the language: Path diagrams and command language
[SEM] Intro 4 — Substantive concepts
[SEM] Builder — SEM Builder
[SEM] Builder, generalized — SEM Builder for generalized models
[SEM] sem and gsem path notation — Command syntax for path diagrams
The structural equation modeling way of describing models is deceptively simple. It is deceptive because the machinery underlying structural equation modeling is sophisticated, complex, and sometimes temperamental, and it can be temperamental both in substantive statistical ways and in practical computer ways.

Professional researchers need to understand these issues.

Remarks and examples

Remarks are presented under the following headings:

- Differences in assumptions between sem and gsem
  - sem: Choice of estimation method
  - gsem: Choice of estimation method
- Treatment of missing values
- Variable types: Observed, latent, endogenous, exogenous, and error
- Constraining parameters
  - Constraining path coefficients to specific values
  - Constraining intercepts to specific values (suppressing the intercept)
  - Constraining path coefficients or intercepts to be equal
  - Constraining covariances to be equal (or to specific values)
  - Constraining variances to specific values (or to be equal)
- Identification 1: Substantive issues
  - Not all models are identified
  - How to count parameters
  - What happens when models are unidentified
  - How to diagnose and fix the problem
- Identification 2: Normalization constraints (anchoring)
  - Why the problem arises
  - How the problem would manifest itself
  - How sem (gsem) solves the problem for you
  - Overriding sem’s (gsem’s) solution
Differences in assumptions between sem and gsem

**sem** fits standard linear SEMs.

**gsem** fits generalized SEMs, by which we mean

1. **gsem** fits not just linear but generalized linear models,
2. **gsem** fits multilevel mixed models,
3. **gsem** fits models with categorical latent variables,
4. **gsem** fits items 1 and 2 combined, and
5. **gsem** fits items 1 and 3 combined.

There is a difference in assumptions between standard linear SEMs and generalized SEMs.

Standard linear SEMs generally assume that the observed endogenous variables, the observed exogenous variables, the latent endogenous variables, and the latent exogenous variables are jointly distributed normally with mean $\mu$ and variance matrix $\Sigma$. In this formulation, we include the $e.$ error variables among the latent exogenous variables.

Generalized SEMs drop the observed variables and categorical latent variables from the joint-normality assumption. Generalized SEMs treat the observed exogenous variables as given and produce estimates conditional on their values. This seems a minor difference, but for some researchers it has important implications.

Consider a researcher wishing to include a subject’s age and age-squared among the observed exogenous variables of a model. Variables age and age-squared simply cannot be jointly normally distributed. Including age and age-squared violates the assumptions of standard linear SEMs, at least officially. You may now suspect that such researchers—perhaps including yourself—should use **gsem** and not **sem**. Because standard linear SEMs are a subset of generalized SEMs, that should not be much of an inconvenience.

It is interesting to note, however, that **sem** and **gsem** produce the same numeric solutions for the parameters and the standard errors when both can fit the same model.

All of which is to say that although it is typical to assume joint normality of all variables when deriving the standard linear SEM, joint normality is not strictly necessary. The lesser assumption of joint normality conditional on the observed exogenous variables is sufficient. Even the normality assumption can be relaxed and replaced with i.i.d., and even the i.i.d. assumption can be relaxed.

We will discuss these issues below.

Even so, some structural equation modeling statistics are dependent on the full joint normality assumption. Many goodness-of-fit tests fall into that category. Throughout this manual, we will be explicit about when the full joint-normality assumption is required.

**gsem** never requires the full joint-normality assumption. Standard linear SEMs have a history that includes the full joint-normality assumption; generalized SEMs have no such history. At the end of the day, both can be considered $M$ estimation.

**sem:** Choice of estimation method

**sem** provides four estimation methods: maximum likelihood (ML; the default), quasimaximum likelihood (QML), asymptotic distribution free (ADF), and maximum likelihood with missing values (MLMV).
Strictly speaking, the assumptions one must make to establish the consistency of the estimates and their asymptotic normality is determined by the method used to estimate them. We want to give you advice on when to use each.

1. **ML** is the method that *sem* uses by default. In *sem*, the function being maximized formally assumes the full joint normality of all the variables, including the observed variables. But the full joint-normality assumption can be relaxed, and the substitute conditional-on-the-observed-exogenous-variables is sufficient to justify all reported estimates and statistics except the log-likelihood value and the model-versus-saturated $\chi^2$ test.

Relaxing the constraint that latent variables outside of the error variables are not normally distributed is more questionable. In the measurement model $X->x_1 x_2 x_3 x_4$, simulations with the violently nonnormal $X \sim \chi^2(2)$ produced good results except for the standard error of the estimated variance of $X$. Note that it was not the coefficient on $X$ that was estimated poorly, it was not the coefficient’s standard error, and it was not even the variance of $X$ that was estimated poorly. It was the standard error of the variance of $X$. Even so, there are no guarantees.

*sem* uses method ML when you specify `method(ml)` or when you omit the `method()` option altogether.

2. QML uses ML to fit the model parameters but relaxes the normality assumptions when estimating the standard errors. QML handles nonnormality by adjusting standard errors.

Concerning the parameter estimates, everything just said about ML applies to QML because those estimates are produced by ML.

Concerning standard errors, we theoretically expect consistent standard errors, and we practically observe that in our simulations. In the measurement model with $X \sim \chi^2(2)$, we even obtained good standard errors of the estimated variance of $X$. QML does not really fix the problem of nonnormality of latent variables, but it does tend to do a better job.

*sem* uses method QML when you specify `method(ml) vce(robust)` or, because `method(ml)` is the default, when you specify just `vce(robust)`.

When you specify `method(ml) vce(sbentler)` or just `vce(sbentler)`, the Satorra–Bentler scaled $\chi^2$ test is reported. When observed data are nonnormal, the standard model-versus-saturated test statistic does not follow a $\chi^2$ distribution. The Satorra–Bentler scaled $\chi^2$ statistic uses a function of fourth-order moments to adjust the standard goodness-of-fit statistic so that it has a mean that more closely follows the reference $\chi^2$ distribution. The corresponding robust standard errors, which are adjusted using a function of fourth-order moments, are reported as well. For details, see Satorra and Bentler (1994).

3. ADF makes no assumption of joint normality or even symmetry, whether for observed or latent variables. Whereas QML handles nonnormality by adjusting standard errors and not point estimates, ADF produces justifiable point estimates and standard errors under nonnormality.

For many researchers, this is most important for relaxing the assumption of normality of the errors, and because of that, ADF is sometimes described that way. ADF in fact relaxes the normality assumption for all latent variables.

Along the same lines, it is sometimes difficult to be certain exactly which normality assumptions are being relaxed when reading other sources. It sometimes seems that ADF uniquely relaxes the assumption of the normality of the observed variables, but that is not true. Other methods, even ML, can handle that problem.

ADF is a form of weighted least squares (WLS). ADF is also a generalized method of moments (GMM) estimator. In simulations of the measurement model with $X \sim \chi^2(2)$, ADF produces...
excellent results, even for the standard error of the variance of $X$. Be aware, however, that ADF is less efficient than ML when latent variables can be assumed to be normally distributed. If latent variables (including errors) are not normally distributed, on the other hand, ADF will produce more efficient estimates than ML or QML.

`sem` uses method ADF when you specify `method(ADF)`.

4. MLMV aims to retrieve as much information as possible from observations containing missing values.

   In this regard, `sem` methods ML, QML, and ADF do a poor job. They are known as listwise deleters. If variable $x_1$ appears someplace in the model and if $x_1$ contains a missing value in observation 10, then observation 10 simply will not be used. This is true whether $x_1$ is endogenous or exogenous and even if $x_1$ appears in some equations but not in others.

   Method MLMV, on the other hand, is not a deleter at all. Observation 10 will be used in making all calculations.

   For method MLMV to perform what might seem like magic, joint normality of all variables is assumed and missing values are assumed to be missing at random (MAR). MAR means either that the missing values are scattered completely at random throughout the data or that values more likely to be missing than others can be predicted by the variables in the model.

   Method MLMV formally requires the assumption of joint normality of all variables, both observed and latent. If your observed variables do not follow a joint normal distribution, you may be better off using ML, QML, or ADF and simply omitting observations with missing values.

   `sem` uses method MLMV when you specify `method(mlmv)`. See [SEM] Example 26.

### gsem: Choice of estimation method

`gsem` provides only two estimation methods: maximum likelihood (ML; the default) and quasi-maximum likelihood (QML).

1. ML is the method that `gsem` uses by default. This is the same ML used by `sem` but applied to a different likelihood function. The `sem` likelihood function assumes and includes joint normality of all variables. The `gsem` likelihood function assumes only conditional normality. Because likelihood functions are different, the likelihood values reported by `sem` and `gsem` are not comparable except in the case where there are no observed exogenous variables.

   ML formally assumes conditional normality, and thus continuous latent variables are still assumed to be normally distributed. What we said in the `sem` case about relaxing the assumption of normality of the latent variables applies equally in the `gsem` case.

   `gsem` uses method ML when you specify `method(ml)` or when you omit the `method()` option altogether.

2. QML uses ML to fit the model but relaxes the conditional normality assumptions when estimating the standard errors. QML handles nonnormality by adjusting standard errors.

   Everything said about `sem`’s QML applies equally to `gsem`’s QML.

   `gsem` uses method QML when you specify `vce(robust)`.

Because the choice of method often affects convergence with `sem`, in the `gsem` case there is a tendency to confuse choice of integration method with maximization method. However, there are no issues related to assumptions about integration method; choice of integration method is purely a mechanical issue. This is discussed in [SEM] Intro 12.
Treatment of missing values

`sem`, `sem` with method MLMV, and `gsem` treat missing values differently.

1. `sem` by default is a listwise deleter.
   If variable x1 appears in the model and if x1 contains missing in observation 10, then observation 10 will not be used. This is true whether x1 is endogenous or exogenous and even if x1 appears in some equations but not in others.

2. `sem` with method MLMV is not a deleter at all; it uses all observations.
   If variable x1 appears in the model and if x1 contains missing in observation 10, then observation 10 will still be used. Doing this formally requires assuming the joint normality of all observed variables and was discussed in item 4 of `sem: Choice of estimation method`.

3. `gsem` by default is an equationwise deleter for models without categorical latent variables.
   The abridged meaning is that `gsem` will often be able to use more observations from the data than `sem` will, assuming you do not use `sem` with method MLMV.
   The full meaning requires some setup. Consider a model of at least five equations. Assume that observed exogenous variable x1 appears in the first equation but not in equations 2–4; that equation 1 predicts y1; that y1 appears as a predictor in equations 2–4; and that x1 contains missing in observation 10.
   If endogenous variable y1 is latent or observed and of family Gaussian, link identity, but without censoring, then
   3.1 Observation 10 will be ignored in making calculations related to equation 1.
   3.2 Observation 10 will also be ignored in making calculations related to equations 2–4 because y1, a function of x1, appears in them.
   3.3 The calculations for the other equation(s) will include observation 10.
   Alternatively, if y1 is observed and not family Gaussian, link identity, or has censoring, then item 3.2 changes:
   3.2 Observation 10 will be used in making calculations related to equations 2–4 even though y1, a function of x1, appears in them.
   As we said at the outset, the result of all of this is that `gsem` often uses more observations than does `sem` (excluding method MLMV).

4. `gsem` has an option, `listwise`, that duplicates the `sem` rules. This is used in testing of Stata. There is no reason you would want to specify the option.

5. `gsem` uses listwise deletion for models with categorical latent variables.

Variable types: Observed, latent, endogenous, exogenous, and error

Structural equation models can contain four different types of variables:

1. observed exogenous
2. latent exogenous
3. observed endogenous
4. latent endogenous
As a software matter, it is useful to think as though there is a fifth type, too:

5. error

Errors are in fact a special case of latent exogenous variables, but there will be good reason to consider them separately.

As a language matter, it is sometimes useful to think of there being yet another type of variable, namely,

6. measure or measurement

Measurement variables are a special case of observed endogenous variables.

Let us explain:

**Observed.**
A variable is observed if it is a variable in your dataset. In this documentation, we often refer to observed variables with \( x_1, x_2, \ldots, y_1, y_2 \), and so on, but in reality observed variables have names such as `mpg`, `weight`, `testscore`, and so on.

**Latent.**
A variable is latent if it is not observed. A variable is latent if it is not in your dataset but you wish it were. You wish you had a variable recording the propensity to commit violent crime, or socioeconomic status, or happiness, or true ability, or even income. Sometimes, latent variables are imagined variants of real variables, variables that are somehow better, such as being measured without error. At the other end of the spectrum are latent variables that are not even conceptually measurable.

In this documentation, latent variables usually have names such as `L1`, `L2`, `F1`, \ldots, but in real life the names are more descriptive such as `VerbalAbility`, `SES`, and so on. The `sem` and `gsem` commands assume that variables are latent if the first letter of the name is capitalized, so we will always capitalize our latent variable names.

**Endogenous.**
A variable is endogenous (determined within the system) if any path points to it.

**Exogenous.**
A variable is exogenous (determined outside the system) if paths only originate from it or, equivalently, no path points to it.

Now that we have the above definitions, we can better understand the five types of variables:

1. **Observed exogenous.**
   A variable in your dataset that is treated as exogenous in your model.

2. **Latent exogenous.**
   An unobserved variable that is treated as exogenous in your model.

3. **Observed endogenous.**
   A variable in your dataset that is treated as endogenous in your model.

4. **Latent endogenous.**
   An unobserved variable that is treated as endogenous in your model.

5. **Error.**
   Mathematically, error variables are just latent exogenous variables. In `sem` and `gsem`, however, errors are different in that they have defaults different from the other latent exogenous variables.

Errors are named `e`. So, for example, the error variable associated with observed endogenous variable \( y_1 \) has the full name `e.y1`; the error variable associated with latent endogenous variable \( L1 \) has the full name `e.L1`. 
In \texttt{sem}, each endogenous variable has a corresponding \texttt{e}. variable.

In \texttt{gsem}, observed endogenous variables associated with family Gaussian have corresponding \texttt{e}. variables, but other observed endogenous variables do not. All continuous latent endogenous variables have an associated \texttt{e}. variable, but that is not a special case because all continuous latent endogenous variables are assumed to be family Gaussian.

In the Builder, when you create an endogenous variable, the variable’s corresponding error variable instantly springs into existence. The same happens in the command language, you just do not see it. In addition, error variables automatically and inalterably have their path coefficient constrained to be 1.

The covariances between the different variable types are given in the table below. The latent variables discussed here are continuous latent variables. In the table,

1. 0 means 0 and there are no options or secret tricks to change the value from 0.
2. \texttt{i} means as implied by your model and beyond your control.
3. (#) means to see the numbered note below the table.

If an entry in the matrix below is 0 or \texttt{i}, then you may not draw curved paths between variables of the specified types.

<table>
<thead>
<tr>
<th></th>
<th>Observed exogenous</th>
<th>Latent exogenous</th>
<th>Observed endogenous</th>
<th>Latent endogenous</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observed exogenous</td>
<td>(1) &amp; (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latent exogenous</td>
<td>(3)</td>
<td>(4) &amp; (5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Observed endogenous</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latent endogenous</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td>(6) &amp; (7)</td>
</tr>
<tr>
<td>Error</td>
<td>0</td>
<td>0</td>
<td>\texttt{i}</td>
<td>\texttt{i}</td>
<td></td>
</tr>
</tbody>
</table>

1. Variances of observed exogenous variables:

1.1 Builder, \texttt{sem} mode: Taken as given, assuming not using method MLMV. Can be estimated or constrained; if so, you are assuming normality of the observed exogenous variables unless using method ADF.

1.2 \texttt{sem} command: Same as item 1.1. Use \texttt{var()} option to estimate or constrain.

1.3 Builder, \texttt{gsem} mode: Taken as given. Cannot be estimated or constrained.

1.4 \texttt{gsem} command: Same as item 1.3.

2. Covariances between observed exogenous variables:

2.1 Builder, \texttt{sem} mode: Taken as given, assuming not using method MLMV. Can be estimated or constrained by drawing curved paths between variables; if so, you are assuming normality of the observed exogenous variables unless using method ADF.

2.2 \texttt{sem} command: Same as item 2.1. Use \texttt{cov()} option to estimate or constrain.

2.3 Builder, \texttt{gsem} mode: Taken as given. Cannot be estimated or constrained. Path may not be drawn.

2.4 \texttt{gsem} command: Same as item 2.3.

3. Covariances between latent exogenous and observed exogenous variables:

3.1 Builder, \texttt{sem} mode: Constrained to be 0 unless a curved path is drawn between variables, in which case it is estimated. Can be constrained.
3.2 \texttt{sem} command: Assumed to be nonzero and therefore estimated by default. Can be constrained (even to 0) using \texttt{cov()} option.

3.3 Builder, \texttt{gsem} mode: Assumed to be nonzero. Cannot be estimated or constrained because this covariance is not among the identified parameters of the generalized SEM.

3.4 \texttt{gsem} command: Same as item 3.3.

4. Variances of latent exogenous variables:
   4.1 Builder, \texttt{sem} mode: Variances are estimated and can be constrained.
   4.2 \texttt{sem} command: Variances are estimated and can be constrained using \texttt{var()} option.
   4.3 Builder, \texttt{gsem} mode: Almost the same as item 4.1 except variances cannot be constrained to 0.
   4.4 \texttt{gsem} command: Almost the same as item 4.2 except variances cannot be constrained to 0.

5. Covariances between latent exogenous variables:
   5.1 Builder, \texttt{sem} mode: Assumed to be 0 unless a curved path is drawn between variables. Path may include constraints.
   5.2 \texttt{sem} command: Assumed to be nonzero and estimated, the same as if a curved path without a constraint were drawn in the Builder. Can be constrained (even to 0) using \texttt{cov()} option.
   5.3 Builder, \texttt{gsem} mode: Same as item 5.1.
   5.4 \texttt{gsem} command: Same as item 5.2.

6. Variances of errors:
   6.1 Builder, \texttt{sem} mode: Estimated. Can be constrained.
   6.2 \texttt{sem} command: Estimated. Can be constrained using \texttt{var()} option.
   6.3 Builder, \texttt{gsem} mode: Almost the same as item 6.1 except variances cannot be constrained to 0.
   6.4 \texttt{gsem} command: Almost the same as item 6.2 except variances cannot be constrained to 0.

7. Covariances between errors:
   7.1 Builder, \texttt{sem} mode: Assumed to be 0. Can be estimated by drawing curved paths between variables. Can be constrained.
   7.2 \texttt{sem} command: Assumed to be 0. Can be estimated or constrained using \texttt{cov()} option.
   7.3 Builder, \texttt{gsem} mode: Almost the same as item 7.1 except covariances between errors cannot be estimated or constrained if one or both of the error terms correspond to a generalized response with family Gaussian, link log, or link identity with censoring.
   7.4 \texttt{gsem} command: Almost the same as item 7.2 except covariances between errors cannot be estimated or constrained if one or both of the error terms correspond to a generalized response with family Gaussian, link log, or link identity with censoring.
The properties of categorical latent variables differ from those of the continuous latent variables that we have discussed to this point. Categorical latent variables cannot be included in models with other types of latent variables. They do not covary by default. To specify a covariance, rather than using the \texttt{cov()} option, you add an intercept or a predictor for a cell of the interaction of the latent variables. For instance, you might type \texttt{2.C\# 3.D <- _cons}. Categorical latent variables do not covary with any other types of variables in the model. Instead, parameters related to other variables are allowed to vary across the classes of the categorical latent variable.

Finally, there is a sixth variable type that we sometimes find convenient to talk about:

\textit{Measure} or \textit{measurement}.

A measure variable is an observed endogenous variable with a path from a latent variable. We introduce the word “measure” not as a computer term or even a formal modeling term but as a convenience for communication. It is a lot easier to say that $x_1$ is a measure of $X$ than to say that $x_1$ is an observed endogenous variable with a path from latent variable $X$ and so, in a real sense, $x_1$ is a measurement of $X$.

In our measurement model,

\begin{figure}
\centering
\includegraphics{measurement_model.png}
\end{figure}

the variables are

- latent exogenous: $X$
- error: $e.x_1, e.x_2, e.x_3, e.x_4$
- observed endogenous: $x_1, x_2, x_3, x_4$

All the observed endogenous variables in this model are measures of $X$. 
Constraining parameters

Constraining path coefficients to specific values

If you wish to constrain a path coefficient to a specific value, you just write the value next to the path. In our measurement model without correlation of the residuals,

we indicate that the coefficients $e.x_1, \ldots, e.x_4$ are constrained to be 1 by placing a small 1 along the path. We can similarly constrain any path in the model.

If we wanted to constrain $\beta_2 = 1$ in the equation

$$x_2 = \alpha_2 + X\beta_2 + e.x_2$$

we would write a 1 along the path between $X$ and $x_2$. If we were instead using sem's or gsem's command language, we would write

$$(x1<-X) (x2<-X@1) (x3<-X) (x4<-X)$$

That is, you type an @ symbol immediately after the variable whose coefficient is being constrained, and then you type the value.

Constraining intercepts to specific values (suppressing the intercept)

Constraining path coefficients is common. Constraining intercepts is less so, and usually when the situation arises, you wish to constrain the intercept to 0, which is often called “suppressing the intercept”.

Although it is unusual to draw the paths corresponding to intercepts in path diagrams, they are assumed, and you could draw them if you wish. A more explicit version of our path diagram for the measurement model is
The path coefficient of _cons to x1 corresponds to $\alpha_1$ in

$$x_1 = \alpha_1 + X\beta_1 + e.x_1$$

and the path coefficient of _cons to x2 corresponds to $\alpha_2$ in

$$x_2 = \alpha_2 + X\beta_2 + e.x_2$$

and so on.
Obviously, if you wanted to constrain a particular intercept to a particular value, you would write the value along the path. To constrain $\alpha_2 = 0$, you could draw

Because intercepts are assumed, you could omit drawing the paths from _cons to x1, _cons to x3, and _cons to x4:
Just as with the Builder, the command language assumes paths from _cons to all endogenous variables, but you could type them if you wished:

\[(x1<-X \_cons) (x2<-X \_cons) (x3<-X \_cons) (x4<-X \_cons)\]

If you wanted to constrain \(\alpha_2 = 0\), you could type

\[(x1<-X \_cons) (x2<-X \_cons@0) (x3<-X \_cons) (x4<-X \_cons)\]

or you could type

\[(x1<-X) (x2<-X \_cons@0) (x3<-X) (x4<-X)\]

### Constraining path coefficients or intercepts to be equal

If you wish to constrain two or more path coefficients to be equal, place a symbolic name along the relevant paths:

\[X\]

\[x1\]

\[\varepsilon_1\]

\[x2\]

\[\varepsilon_2\]

\[x3\]

\[\varepsilon_3\]

\[x4\]

\[\varepsilon_4\]

\[myb \quad myb\]

In the diagram above, we constrain \(\beta_2 = \beta_3\) because we stated that \(\beta_2 = myb\) and \(\beta_3 = myb\).

You follow the same approach in the command language:

\[(x1<-X) (x2<-X@myb) (x3<-X@myb) (x4<-X)\]

This works the same way with intercepts. Intercepts are just paths from _cons, so to constrain intercepts to be equal, you add symbolic names to their paths. In the command language, you constrain \(\alpha_1 = \alpha_2\) by typing

\[(x1<-X \_cons@c) (x2<-X \_cons@c) (x3<-X) (x4<-X)\]

See [SEM] Example 8.

### Constraining covariances to be equal (or to specific values)

If you wish to constrain covariances, usually you will want to constrain them to be equal instead of to a specific value. If we wanted to fit our measurement model and allow correlation between e.x2 and e.x3 and between e.x3 and e.x4, and we wanted to constrain the covariances to be equal, we could draw
If you instead wanted to constrain the covariances to specific values, you would place the value along the paths in place of the symbolic names.

In the command language, covariances (curved paths) are specified using the `cov()` option. To allow covariances between `e.x2` and `e.x3` and between `e.x3` and `e.x4`, you would type

\[
(x1<-X) (x2<-X) (x3<-X) (x4<-X), \text{cov}(e.x2*e.x3) \text{cov}(e.x3*e.x4)
\]

To constrain the covariances to be equal, you would type

\[
(x1<-X) (x2<-X) (x3<-X) (x4<-X), \text{cov}(e.x2*e.x3@myc) \text{cov}(e.x3*e.x4@myc)
\]

### Constraining variances to specific values (or to be equal)

Variances are like covariances except that in path diagrams drawn by some authors, variances curve back on themselves. In the Builder, variances appear inside or beside the box or circle. Regardless of how they appear, variances may be constrained to normalize latent variables, although normalization is handled by `sem` and `gsem` automatically (something we will explain in *How sem (gsem) solves the problem for you* under *Identification 2: Normalization constraints (anchoring)* below).

In the Builder, you constrain variances by clicking on the variable and using the lock box to specify the value, which can be a number or a symbol. In the command language, variances are specified using the `var()` option as we will explain below.
Let’s assume that you want to normalize the latent variable $X$ by constraining its variances to be 1. You do that by drawing

![Diagram](image)

In the command language, we specify this model as

$$(x1<-X) (x2<-X) (x3<-X) (x4<-X), \text{var}(X@1)$$

Constraining latent exogenous variables to have unit variance as an identifying restriction may be desirable when you wish simultaneously to constrain their correlations with other latent exogenous variables. `SEM` and `GSEM` allow you to constrain covariances, not correlations. Covariances are equal to correlations when variances are 1.

In more complicated models, you may wish to constrain variances of latent exogenous variables to be equal. You can do that by specifying a symbolic name.

**Identification 1: Substantive issues**

**Not all models are identified**

Just because you can draw the path diagram for a model, write its equations, or write it in Stata’s command syntax, does not mean the model is identified. Identification refers to the conceptual constraints on parameters of a model that are required for the model’s remaining parameters to have a unique solution. A model is said to be unidentified if these constraints are not supplied. These constraints are of two types: substantive constraints and normalization constraints. We will begin by discussing substantive constraints because that is your responsibility; the software provides normalization constraints automatically.

**How to count parameters**

If you are fitting a model with `SEM` and your model has $K$ observed variables, then your data contain $K(K+1)/2$ second-order moments, and thus $p$, the number of parameters based on second-order moments that can be estimated, cannot exceed $K(K+1)/2$.

Every path in your model contributes 1 to $p$ unless the parameter is constrained to a specific value, and then it does not contribute at all. If two parameters are constrained to be equal, the two parameters count as one. In counting $p$, you must remember to count the curved paths from latent variables back to themselves, which is to say, the variances. Just counting the number of parameters can be challenging. And even if $p \le K(K+1)/2$, your model may not be identified. Identification depends not only on the number of paths but also on their locations.
Counting parameters can be even more difficult in the case of certain generalized linear (\textit{gsem}) models. For a discussion of this, see Skrondal and Rabe-Hesketh (2004, chap. 5).

Even in the non-\textit{gsem} case, books have been written on this subject, and we will refer you to them. A few are Bollen (1989), Brown (2015), Kline (2016), and Kenny (1979). We will refer you to them, but do not be surprised if they refer you back to us. Brown (2015, 179) writes, “Because latent variable software programs are capable of evaluating whether a given model is identified, it is often most practical to simply try to estimate the solution and let the computer determine the model’s identification status.” That is not bad advice.

\textbf{What happens when models are unidentified}

So what happens when you attempt to fit an unidentified model? In some cases, \texttt{sem (gsem)} will tell you that your model is unidentified. If your model is unidentified for subtle substantive reasons, however, you will see

\begin{verbatim}
initial values not feasible
r(1400);
\end{verbatim}

or

\begin{verbatim}
Iteration 50: log likelihood = -337504.44 (not concave)
Iteration 51: log likelihood = -337504.44 (not concave)
Iteration 52: log likelihood = -337504.44 (not concave)
.
.
.
Iteration 101: log likelihood = -337504.44 (not concave)
.
.
.
\end{verbatim}

In the latter case, \texttt{sem (gsem)} will iterate forever, reporting the same criterion value (such as log likelihood) and saying “not concave” over and over again.

Observing periods of the “not concave” message is not concerning, so do not overreact at the first occurrence. Become concerned when you see “not concave” and the criterion value is not changing, and even then, stay calm for a short time because the value might be changing in digits you are not seeing. If the iteration log continues to report the same value several times, however, press \texttt{Break}. Your model is probably not identified.

\textbf{How to diagnose and fix the problem}

You must find and fix the problem.

If you get the “initial values not feasible” message, your goal is to find feasible initial values and thereby either solve the problem or convert it into the infinitely long iteration log problem; see [SEM] Intro 12.

If you get the infinitely long log, rerun the model specifying \texttt{sem’s (gsem’s) iterate(#) option, where # is large enough to reach the “not concave” messages with constant criterion value. \texttt{sem (gsem)} will iterate that many times and then report the results it has at that point. Look at the output for missing standard errors. Those parameters are unidentified, and you need to think about either changing your model so that they become identified or placing constraints on them.
Normalization constraints (anchoring) are provided automatically by `sem` and `gsem`. It is rare indeed that the automatically provided constraints will be the source of convergence problems.

Models with continuous latent variables require normalization constraints because these latent variables have no natural scale. If constraints were not provided, the model would appear to the software the same as a model with a substantive lack of identification; the estimation routine would iterate forever and never arrive at a solution.

The `sem` and `gsem` commands automatically provide normalization constraints to prevent that from happening.

Below we explain why the normalization constraints are required, which normalization constraints `sem` and `gsem` automatically supply, how to override those automatic choices, and how to substitute your own constraints should you desire.

**Why the problem arises**

Imagine a latent variable for propensity to be violent. Your imagination might supply a scale that ranges from 0 to 1 or 1 to 100 or over other values, but regardless, the scale you imagine is arbitrary in that one scale works as well as another.

Scales have two components: mean and variance. If you imagine a latent variable with mean 0 and your colleague imagines the same variable with mean 100, the difference can be accommodated in the parameter estimates by an intercept changing by 100. If you imagine a standard deviation of 1 (variance $1^2 = 1$) and your colleague imagines a standard deviation of 10 (variance $10^2 = 100$), the difference can be accommodated by a path coefficient differing by a multiplicative factor of 10. You might measure an effect as being 1.1 and then your colleague would measure the same effect as being 0.11, but either way you both will come to the same substantive conclusions.

**How the problem would manifest itself**

The problem is that different scales all work equally well, and the software will iterate forever, jumping from one scale to another.

Another way of saying that the means and variances of latent variables are arbitrary is to say that they are unidentified. That’s important because if you do not specify the scale you have in mind, results of estimation will look just like substantive lack of identification.

`sem` (`gsem`) will iterate forever and never arrive at a solution.

**How `sem` (`gsem`) solves the problem for you**

You usually do not need to worry about this problem because `sem` (`gsem`) solves it for you. `sem` (`gsem`) solves the unidentified scale problem for continuous latent variables by

1. Assuming that latent exogenous variables have mean 0.
2. Assuming that latent endogenous variables have intercept 0.
3. Setting the coefficients on paths from latent variables to the first observed endogenous variable to be 1.
4. Setting the coefficients on paths from latent variables to the first latent endogenous variable to be 1 if rule 3 does not apply—if the latent variable is measured by other latent variables only.
Rules 3 and 4 are also known as the unit-loading rules. The variable to which the path coefficient is set to 1 is said to be the anchor for the latent variable.

Applying those rules to our measurement model, when we type

\[(X\rightarrow x_1) (X\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4)\]

\texttt{sem} (\texttt{gsem}) acts as if we typed

\[(X\@1\rightarrow x_1) (X\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4), \text{means}(X\@0)\]

The above four rules are sufficient to provide a scale for latent variables for all models.

**Overriding \texttt{sem}'s (\texttt{gsem}'s) solution**

\texttt{sem} (\texttt{gsem}) automatically applies rules 1 through 4 to produce normalization constraints. There are, however, other normalization constraints that would work as well. In what follows, we will assume that you are well versed in deriving normalization constraints and just want to know how to bend \texttt{sem} (\texttt{gsem}) to your will.

Before you do this, however, let us warn you that substituting your normalization rules for the defaults can result in more iterations being required to fit your model. Yes, one set of normalization constraints are as good as the next, but \texttt{sem}'s (\texttt{gsem})'s starting values are based on its default normalization rules, which means that when you substitute your rules for the defaults, the required number of iterations sometimes increases.

Let’s return to the measurement model:

\[(X\rightarrow x_1) (X\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4)\]

As we said previously, type the above and \texttt{sem} (\texttt{gsem}) acts as if you typed

\[(X\@1\rightarrow x_1) (X\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4), \text{means}(X\@0)\]

If you wanted to assume instead that the mean of \(X\) is 100, you could type

\[(X\rightarrow x_1) (X\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4), \text{means}(X\@100)\]

The \texttt{means()} option allows you to specify mean constraints, and you may do so for latent or observed variables.

Let’s leave the mean at 0 and specify that we instead want to constrain the second path coefficient to be 1:

\[(X\rightarrow x_1) (X\@1\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4)\]

We did not have to tell \texttt{sem} (\texttt{gsem}) not to constrain \(X\rightarrow x_1\) to have coefficient 1. We just specified that we wanted to constrain \(X\rightarrow x_2\) to have coefficient 1. \texttt{sem} (\texttt{gsem}) takes all the constraints that you specify and then adds whatever normalization constraints are needed to identify the model. If what you have specified is sufficient, \texttt{sem} (\texttt{gsem}) does not add its constraints to yours.

Obviously, if we wanted to constrain the mean to be 100 and the second rather than the first path coefficient to be 1, we would type

\[(X\rightarrow x_1) (X\@1\rightarrow x_2) (X\rightarrow x_3) (X\rightarrow x_4), \text{means}(X\@100)\]
If we wanted to constrain the standard deviation of \( X \) to be 10 instead of constraining a path coefficient to be 1, we would type

\[
(X \rightarrow x_1) \ (X \rightarrow x_2) \ (X \rightarrow x_3) \ (X \rightarrow x_4), \quad \text{means(X@100)} \quad \text{var(X@100)}
\]

Standard deviations are specified in variance units when you use the \texttt{var()} option.

References


Also see

- [SEM] Intro 3 — Learning the language: Factor-variable notation (gsem only)
- [SEM] Intro 5 — Tour of models
- [SEM] sem and gsem path notation — Command syntax for path diagrams
- [SEM] sem and gsem option covstructure() — Specifying covariance restrictions
Below is a sampling of SEMs that can be fit by `sem` or `gsem`.

If you have not read [SEM] Intro 2, please do so. You need to speak the language. We also recommend reading [SEM] Intro 4, but that is not required.

Now that you speak the language, we can start all over again and take a look at some of the classic models that `sem` and `gsem` can fit.

Remarks are presented under the following headings:

- Single-factor measurement models
- Item response theory (IRT) models
- Multiple-factor measurement models
- Confirmatory factor analysis (CFA) models
- Structural models 1: Linear regression
- Structural models 2: Gamma regression
- Structural models 3: Binary-outcome models
- Structural models 4: Count models
- Structural models 5: Ordinal models
- Structural models 6: Multinomial logistic regression
- Structural models 7: Survival models
- Structural models 8: Dependencies between response variables
- Structural models 9: Unobserved inputs, outputs, or both
- Structural models 10: MIMIC models
- Structural models 11: Seemingly unrelated regression (SUR)
- Structural models 12: Multivariate regression
- Structural models 13: Mediation models
- Correlations
- Higher-order CFA models
- Correlated uniqueness model
- Latent growth models
- Models with reliability
- Multilevel mixed-effects models
- Latent class models
- Finite mixture models
Single-factor measurement models

A single-factor measurement model is

\[ X \]

\[ x1 \]
\[ x2 \]
\[ x3 \]
\[ x4 \]

The model can be written in Stata command language as

\[(x1<-X) (x2<-X) (x3<-X) (x4<-X)\]

or as

\[(x1 x2 x3 x4<-X)\]

or as

\[(X->x1 x2 x3 x4)\]

or in other ways. All the equivalent ways really are equivalent; no subtle differences will subsequently arise according to your choice.

The measurement model plays an important role in many other SEMs dealing with the observed inputs and the observed outputs:
Because the measurement model is so often joined with other models, it is common to refer to the coefficients on the paths from latent variables to observable endogenous variables as the measurement coefficients and to refer to their intercepts as the measurement intercepts. The intercepts are usually not shown in path diagrams. The other coefficients and intercepts are those not related to the measurement issue.

The measurement coefficients are often referred to as loadings.

This model can be fit by \texttt{sem} or \texttt{gsem}. Use \texttt{sem} for standard linear models (standard means single level); use \texttt{gsem} when you are fitting a multilevel model or when the response variables are generalized linear such as probit, logit, multinomial logit, Poisson, and so on.

See the following examples:

1. \texttt{SEM} Example 1. Single-factor measurement model.
2. \texttt{SEM} Example 27g. Single-factor measurement model (generalized response).
3. \texttt{SEM} Example 30g. Two-level measurement model (multilevel, generalized response).
4. \texttt{SEM} Example 35g. Ordered probit and ordered logit.

\section*{Item response theory (IRT) models}

Item response theory (IRT) models look like the following:

\begin{center}
\begin{tikzpicture}
  \node (L) at (0,0) {$L$};
  \node (item1) at (-2,-1) {Bernoulli \texttt{logit}};
  \node (item2) at (0,-1) {Bernoulli \texttt{logit}};
  \node (item3) at (2,-1) {Bernoulli \texttt{logit}};
  \node (item4) at (4,-1) {Bernoulli \texttt{logit}};
  \draw (L) -- (item1);
  \draw (L) -- (item2);
  \draw (L) -- (item3);
  \draw (L) -- (item4);
\end{tikzpicture}
\end{center}

The items are the observed variables, and each has a 0/1 outcome measuring a latent variable. Often, the latent variable represents ability. These days, it is traditional to fit IRT models using logistic regression, but in the past, probit was used and they were called normal ogive models.

In one-parameter logistic models, also known as 1-PL models and Rasch models, constraints are placed on the paths and perhaps the variance of the latent variable. Either path coefficients are constrained to 1 or path coefficients are constrained to be equal and the variance of the latent variable is constrained to be 1. Either way, this results in the negative of the intercepts of the fitted model being a measure of difficulty.

1-PL and Rasch models can be fit treating the latent variable—ability—as either fixed or random. Abilities are treated as random with \texttt{gsem}.

In two-parameter logistic models (2-PL), no constraints are imposed beyond the one required to identify the latent variable, which is usually done by constraining the variance to 1. This results in path coefficients measuring discriminating ability of the items, and difficulty is measured by the negative of the intercepts divided by the corresponding (slope) coefficient.
IRT has been extended beyond 1-PL and 2-PL models, including extension to other types of generalized responses.

IRT models, including the extensions, can be fit by `gsem`.

See the following examples:

1. [SEM] Example 28g. One-parameter logistic IRT (Rasch) model.
2. [SEM] Example 29g. Two-parameter logistic IRT model.

### Multiple-factor measurement models

A two-factor measurement model is two one-factor measurement models with possible correlation between the factors:

![Diagram of a two-factor measurement model](attachment:image.png)

To obtain a correlation between F1 and F2, we drew a curved path.

The model can be written in Stata command language as

```stata
(F1->x1) (F1->x2) (F1->x3) (F2->x4) (F2->x5) (F2->x6)
```

In the command language, we do not have to include the `cov(F1*F2)` option because, by default, `sem` assumes that exogenous latent variables are correlated with each other. This model can also be written in any of the following ways:

1. `(F1->x1 x2 x3) (F2->x4 x5 x6)`
2. `(x1 x2 x3<-F1) (x4 x5 x6<-F2)`
3. `(x1<-F1) (x2<-F1) (x3<-F1) (x4<-F2) (x5<-F2) (x6<-F2)`

Two-factor measurement models can be fit by `sem` and `gsem`.

See the following examples:

2. [SEM] Example 31g. Two-factor measurement model (generalized response).
Confirmatory factor analysis (CFA) models

The measurement models just shown are also known as confirmatory factor analysis (CFA) models because they can be analyzed using CFA.

In the single-factor model, after estimation, you might want to test that all the indicators have significant loadings by using `test`; see [SEM] `test`. You might also want to test whether the correlations between the errors should have been included in the model by using `estat mindices`; see [SEM] `estat mindices`.

In the multiple-factor measurement model, you might want to test that any of the omitted paths should in fact be included in the model. The omitted paths in the two-factor measurement model above were $F_1 \to x_4$, $F_1 \to x_5$, $F_1 \to x_6$, and $F_2 \to x_1$, $F_2 \to x_2$, $F_2 \to x_3$. `estat mindices` will perform these tests.

These tests are available after `sem` only. CFA is just a measurement model and can be fit by both `sem` and `gsem`.

We show other types of CFA models below.

See the following example:


Structural models 1: Linear regression

Different authors define the meaning of structural models in different ways. Bollen (1989, 4) defines a structural model as the parameters being not of a descriptive nature of association but instead of a causal nature. By that definition, the measurement models above could be structural models, and so could the linear regression below.

Others define structural models as models having paths reflecting causal dependencies between endogenous variables, which would thus exclude the measurement model and linear regression. We will show you a “true” structural model in the next example.

An example of a linear regression is

$$y \leftarrow x_1 \ x_2 \ x_3$$

This model can be written in Stata command language as

$$(y<-x_1 \ x_2 \ x_3)$$

When you estimate a linear regression by using `sem`, you obtain the same point estimates as you would with `regress` and the same standard errors up to a degree-of-freedom adjustment applied by `regress`.
Linear regression models can be fit by \texttt{sem} and \texttt{gsem}. \texttt{gsem} also has options for censoring.

See the following examples:

2. [SEM] Example 38g. Random-intercept and random-slope models (multilevel).
3. [SEM] Example 40g. Crossed models (multilevel).
4. [SEM] Example 43g. Tobit regression.
5. [SEM] Example 44g. Interval regression.

**Structural models 2: Gamma regression**

Gamma regression, also known as log-gamma regression, is used when a continuous outcome is nonnegative, when it ranges from zero to infinity, and often with positively skewed data. It is appropriate when the error variance can be assumed to increase with the mean. Gamma regression gives similar results to linear regression with a logged dependent variable.

\[
\begin{align*}
  y & \sim \text{Gamma}(\log(x_1), \log(x_2), \log(x_3)) \\
\end{align*}
\]

Gamma regression is fit by \texttt{gsem}; specify shorthand \texttt{gamma} or specify \texttt{family(gamma) link(log)}.

You can fit exponential regressions using gamma regression if you constrain the log of the scale parameter to be 0; see [SEM] \texttt{gsem family-and-link options}.

**Structural models 3: Binary-outcome models**

Binary-outcome models have 0/1 response variables. These models include logistic regression (also known as logit), probit, and complementary log-log (also known as clog-log) models.

A simple logistic regression model is

\[
\begin{align*}
  y & \sim \text{Bernoulli}(\text{logit}(x_1), \text{logit}(x_2), \text{logit}(x_3)) \\
\end{align*}
\]
which in command syntax can be written as

\[(y<-x_1\ x_2\ x_3, \text{logit})\]

For the other binary-outcome models, all that changes in the diagram is the names of the family and link; in the command language, the option name changes; see [SEM] \text{gsem\ family-and-link\ options}.

Usually, the observations in binary-outcome data record whether the event occurred, but the data can instead record the number of events and the number of trials by changing the family from Bernoulli to binomial; see [SEM] \text{gsem\ family-and-link\ options}.

Binary-outcome models can be fit by \text{gsem}.

See the following examples:
1. [SEM] \text{Example 33g}. Logistic regression.
2. [SEM] \text{Example 27g}. Single-factor measurement model (generalized response).
3. [SEM] \text{Example 34g}. Combined models (generalized responses).

\textbf{Structural models 4: Count models}

Count models have response variables that are counts of things or events. These models include Poisson and negative binomial models. A simple example of a Poisson model is

\[
y \sim \text{Poisson}(x_1\ x_2\ x_3)
\]

which in command syntax can be written as

\[(y<-x_1\ x_2\ x_3, \text{poisson})\]

For negative binomial models, in path diagrams, the family changes to \text{nbreg} and, in the command syntax, the option changes to \text{nbreg}.

In Poisson models, both the mean and the variance are determined by a single parameter: the rate at which the event occurs. One use of negative binomial models is to handle overdispersion, which is when the variance is greater than the variance that would be predicted by a Poisson model.

The way we have diagrammed the model and written the command, we are assuming that each observation was at risk for the same length of time. If observations varied in this, the command would be

\[(y<-x_1\ x_2\ x_3, \text{poisson\ exposure(etime)})\]

where variable \text{etime} records each observation’s exposure time. The diagram would not change, but we would nonetheless enter the exposure time into the Builder. See [SEM] \text{gsem\ family-and-link\ options}. 

\[
y \sim \text{Poisson}(x_1\ x_2\ x_3)
\]
Count models are fit by \texttt{gsem}.

See the following examples:

1. \texttt{SEM} \textbf{Example 34g}. Combined models (generalized responses).
2. \texttt{SEM} \textbf{Example 39g}. Three-level model (multilevel, generalized response).

\section*{Structural models 5: Ordinal models}

Ordinal models have two or more possible outcomes that are ordered, such as responses of the form “a little”, “average”, and “a lot”; or “strongly disagree”, “disagree”, . . . , “strongly agree”. The outcomes are usually numbered 1, 2, . . . , \(k\). These models include ordered probit, ordered logit, and ordered complementary log-log (also known as ordered clog-log). A simple example of an ordered probit model is

\begin{equation}
\begin{align*}
\text{y} & \sim \text{oprobit}(\text{x}_1, \text{x}_2, \text{x}_3) \\
\end{align*}
\end{equation}

which in command syntax can be written as

\begin{verbatim}
(y<-x1 x2 x3, oprobit)
\end{verbatim}

All that changes for the other models is the link name that appears in the path diagram or the option that appears in the command language.

Ordinal models are fit by \texttt{gsem}.

See the following examples:

1. \texttt{SEM} \textbf{Example 35g}. Ordered probit and ordered logit.
2. \texttt{SEM} \textbf{Example 31g}. Two-factor measurement model (generalized response).
3. \texttt{SEM} \textbf{Example 32g}. Full structural equation model (generalized response).
4. \texttt{SEM} \textbf{Example 36g}. MIMIC model (generalized response).

\section*{Structural models 6: Multinominal logistic regression}

Multinomial logistic regression, also known as multinomial logit, is similar to ordinal models in that it, too, deals with multiple responses; however, in multinomial logistic regression, the responses cannot be ordered. Examples of multinomial logit response include method of transportation (car, public transportation, etc.) or ice-cream flavor (vanilla, chocolate, etc.).

Just as with ordinal models, the outcome is usually recorded as a single variable containing 1, 2, . . . , \(k\), but in path diagrams, factor-variable notation is used to identify the outcomes:
In command syntax, this can be written as

```
(i.y<-x1 x2, mlogit)
```

In the example above, we have paths from all predictor variables to all outcomes. That is common but not required.

The multinomial logistic regression model is fit by `gsem`.

See the following examples:

1. [SEM] Example 37g. Multinomial logistic regression.
2. [SEM] Example 41g. Two-level multinomial logistic regression (multilevel).

**Structural models 7: Survival models**

Survival models are fit to response variables that measure survival times. The data may also contain observations for which a failure is not observed. For those observations, our response variable represents the time until the observation is censored.

Parametric survival models may be fit using a variety of distributions. A simple example of a Weibull survival model is

```
(time<-x1 x2 x3, family(weibull))
```
If some of the observations were censored, the command would be

\[(\text{time}<-x1\ x2\ x3,\ \text{family(weibull, failure(fail))))\]

where fail is an indicator variable coded as 1 for observations that failed and 0 for observations that were censored. The diagram would not change, but we would nonetheless specify fail as the failure indicator by using the Builder.

`gsem` fits parametric survival models using exponential, Weibull, gamma, loglogistic, and lognormal distributions. See [SEM] `gsem family-and-link options` for details of all options available when fitting survival models, including those for specifying the survival distribution and censoring.

See the following examples:

1. [SEM] Example 47g. Exponential survival model.
2. [SEM] Example 48g. Loglogistic survival model with censored and truncated data.
3. [SEM] Example 49g. Multiple-group Weibull survival model.

**Structural models 8: Dependencies between response variables**

An example of a structural model having paths between response (endogenous) variables is

![Diagram]

This model can be written in Stata command language as

\[(y1<-x1\ x2\ x3\ x4)\ (y2<-y1\ x2\ x3)\]
In this example, all inputs and outputs are observed and the errors are assumed to be uncorrelated. In these kinds of models, it is common to allow correlation between errors:

\[ y_1 \leftarrow x_1, x_2, x_3, x_4 \]  
\[ y_2 \leftarrow y_1, x_2, x_3 \]  
\[ \text{cov}(e_{y1}, e_{y2}) \]

The model above can be written in Stata command language as

\[(y_1<-x_1 \ x_2 \ x_3 \ x_4) \ (y_2<-y_1 \ x_2 \ x_3), \text{cov}(e_{y1}*e_{y2})\]

This structural model is said to be overidentified. If we omitted \(y_1 \leftarrow x_4\), the model would be just-identified. If we also omitted \(y_1 \leftarrow x_1\), the model would be unidentified.

When you fit the above model using `sem`, you obtain slightly different results from those you would obtain with `ivregress liml`. This is because `sem` with default `method(ml)` produces full-information maximum likelihood rather than limited-information maximum likelihood results.

Analysis of models for observed variables that include dependencies between endogenous variables may also be referred to as path analysis. Acock (2013, chap. 2) discusses path analysis with `sem` in more detail.

These models can be fit by `sem` or `gsem`. When using `gsem` to fit models with generalized response variables, non-Gaussian responses and Gaussian responses with the log link, or censoring, can only be included in recursive portions of models.

See the following examples:

2. [SEM] Example 34g. Combined models (generalized responses).
3. [SEM] Example 42g. One- and two-level mediation models (multilevel).
4. [SEM] Example 43g. Tobit regression.
5. [SEM] Example 44g. Interval regression.
7. [SEM] Example 46g. Endogenous treatment-effects model.
Structural models 9: Unobserved inputs, outputs, or both

Perhaps in a structural model such as

\[
\begin{align*}
\varepsilon_1 & \quad \varepsilon_2 \\
\text{y}_1 & \quad \text{y}_2 \\
\text{x}_1 & \quad \text{x}_2 & \quad \text{x}_3 \\
\end{align*}
\]

the inputs \( x_1, x_2, \) and \( x_3 \) are concepts and thus are not observed. Assume that we have measurements for them. We can join this structural model example with a three-factor measurement model:

\[
\begin{align*}
\varepsilon_1 & \quad \varepsilon_2 & \quad \varepsilon_3 & \quad \varepsilon_4 & \quad \varepsilon_5 & \quad \varepsilon_6 & \quad \varepsilon_7 & \quad \varepsilon_8 \\
\text{y}_1 & \quad \text{y}_2 \\
\text{z}_1 & \quad \text{z}_2 & \quad \text{z}_3 & \quad \text{z}_4 & \quad \text{z}_5 & \quad \text{z}_6 & \quad \text{z}_7 & \quad \text{z}_8 \\
\end{align*}
\]

Note the curved arrows denoting correlation between the pairs of \( X_1, X_2, \) and \( X_3 \). In the previous path diagram, we had no such arrows between the variables, yet we were still assuming that they were there. In SEM’s path diagrams, correlation between exogenous observed variables is assumed and need not be explicitly shown. When we changed observed variables \( x_1, x_2, \) and \( x_3 \) to be the latent variables \( X_1, X_2, \) and \( X_3 \), we needed to show explicitly the correlations we were allowing. Correlation between latent variables is not assumed and must be shown.
This model can be written in Stata command syntax as follows:

\[
(y_1 <- X_1 \ X_2) \ (y_2 <- y_1 \ X_2 \ X_3) \\
(X_1 -> z_1 \ z_2 \ z_3) \\
(X_2 -> z_4 \ z_5) \\
(X_3 -> z_6 \ z_7 \ z_8), \\
cov(e.y1*e.y2)
\]

We did not include the `cov(X1*X2 \ X1*X3 \ X2*X3)` option, although we could have. In the command language, exogenous latent variables are assumed to be correlated with each other. If we did not want \(X_2\) and \(X_3\) to be correlated, we would need to include the `cov(X2*X3@0)` option.

We changed \(x_1\), \(x_2\), and \(x_3\) to be \(X_1\), \(X_2\), and \(X_3\). In command syntax, variables beginning with a capital letter are assumed to be latent. Alternatively, we could have left the names in lowercase and specified the identities of the latent variables:

\[
(y_1 <- x_1 \ x_2) \ (y_2 <- y_1 \ x_2 \ x_3) \\
(x_1 -> z_1 \ z_2 \ z_3) \\
(x_2 -> z_4 \ z_5) \\
(x_3 -> z_6 \ z_7 \ z_8), \\
cov(e.y1*e.y2) \\
literal{latent(x1 x2 x3)}
\]

Just as we have joined an observed structural model to a measurement model to handle unobserved inputs, we could join the above model to a measurement model to handle unobserved \(y_1\) and \(y_2\).

Models with unobserved inputs, outputs, or both can be fit by `sem` and `gsem`.

See the following examples:

2. [SEM] Example 32g. Full structural equation model (generalized response).
4. [SEM] Example 45g. Heckman selection model.
5. [SEM] Example 46g. Endogenous treatment-effects model.

**Structural models 10: MIMIC models**

MIMIC stands for multiple indicators and multiple causes. An example of a MIMIC model is

![MIMIC Model Diagram]

In this model, the observed causes \(c_1\), \(c_2\), and \(c_3\) determine latent variable \(L\), and \(L\) in turn determines the observed indicators \(i_1\), \(i_2\), and \(i_3\).
This model can be written in Stata command syntax as

\[(i1 \ i2 \ i3<-L) \ (L<-c1 \ c2 \ c3)\]

MIMIC models can be fit by `sem` and `gsem`.

See the following examples:

1. [SEM] Example 10. MIMIC model.
2. [SEM] Example 36g. MIMIC model (generalized response).

**Structural models 11: Seemingly unrelated regression (SUR)**

Seemingly unrelated regression (SUR) is like having two or more separate linear regressions but allowing the errors to be correlated.

An example of an SUR model is

![Diagram of an SUR model](image)

The model above can be written in Stata command syntax as

\[(y1<-x1 \ x2 \ x3) \ (y2<-x3 \ x4), \ cov(e.y1*e.y2)\]

In this example, the two regressions share a common exogenous variable, \(x3\). They do not have to share a common variable, or they could share more variables. If they shared all variables, results would be the same as estimating multivariate regression, shown in the next example.

When you estimate an SUR with `sem`, you obtain the same point estimates as you would with `sureg` if you specify `sureg`'s `isure` option, which causes `sureg` to iterate until it obtains the maximum likelihood result. Standard errors will be different. If the model has exogenous variables only on the right-hand side, then standard errors will be asymptotically identical and, although the standard errors are different in finite samples, there is no reason to prefer one set over the other. If the model being fit is recursive, standard errors produced by `sem` are better than those from `sureg`, both asymptotically and in finite samples.

SUR models can be fit by `sem` and `gsem`. `gsem` will allow you to generalize the model to multilevel. In cases other than family Gaussian, link identity, you can sometimes introduce latent variables to create a similar effect.

See the following example:

Structural models 12: Multivariate regression

A multivariate regression is just an SUR where the different dependent variables share the same exogenous variables:

\[
x_1 \quad x_2 \quad x_3 \\
y_1 \quad \varepsilon_1 \\
y_2 \quad \varepsilon_2
\]

The model above can be written in Stata command syntax as

\[(y_1 \ y_2<-x_1 \ x_2 \ x_3), \text{cov}(e.y_1* e.y_2)\]

When you estimate a multivariate regression with `sem`, you obtain the same point estimates as you would with `mvreg` and the same standard errors up to a multiplicative \(\sqrt{(N - p - 1)/N}\) degree-of-freedom adjustment applied by `mvreg`.

Multivariate regression can be fit by `sem` and `gsem`. `gsem` will allow you to generalize the model to multilevel. In cases other than family Gaussian, link identity, you can sometimes introduce latent variables to create a similar effect.

See the following example:


Structural models 13: Mediation models

Mediation models concern effects that pass through (are mediated by) other variables. Consider response variable \(y\) affected by \(x\) mediated through \(m\):

This can also be specified in command syntax as

\[(y<-m \ x) \ (m<-x)\]
In this simple model, $x$ has a direct effect on $y$ and an indirect (mediated through $m$) effect. The direct effect may be reasonable given the situation, or it may be included just so one can test whether the direct effect is present. If both the direct and indirect effects are significant, the effect of $x$ is said to be partially mediated through $m$.

There are one-level mediation models and various two-level models, and lots of other variations, too.

`sem` and `gsem` can both fit one-level linear models, but you will be better off using `sem`. `gsem` can fit one-level generalized linear models and fit two-level (and higher) models, generalized linear or not.

See the following example:

1. [SEM] Example 42g. One- and two-level mediation models (multilevel).

**Correlations**

We are all familiar with correlation matrices of observed variables, such as

\[
\begin{array}{ccc}
x1 & x2 & x3 \\
x1 & 1.0000 & \\
x2 & 0.7700 & 1.0000 \\
x3 & -0.0177 & -0.2229 & 1.0000 \\
\end{array}
\]

or covariances matrices, such as

\[
\begin{array}{ccc}
x1 & x2 & x3 \\
x1 & 662.172 & \\
x2 & 62.5157 & 9.95558 \\
x3 & -0.769312 & -1.19118 & 2.86775 \\
\end{array}
\]

These results can be obtained from `sem`. The path diagram for the model is

![Path Diagram]

We could just as well leave off the curved paths because `sem` assumes them among observed exogenous variables:

![Path Diagram]

Either way, this model can be written in Stata command syntax as

\[
(\leftarrow x1 \ x2 \ x3)
\]

That is, we simply omit specifying the target of the path, the endogenous variable.
If we fit the model, we will obtain the covariance matrix by default. `correlate` with the `covariance` option produces covariances that are divided by \( N - 1 \) rather than by \( N \). To match this covariance exactly, you need to specify the `nm1` option, which we can do in the command language by typing

\[
(\leftarrow x1 \ x2 \ x3), \ nm1
\]

If we want correlations rather than covariances, we ask for them by specifying the `standardized` option:

\[
(\leftarrow x1 \ x2 \ x3), \ nm1 \ standardized
\]

An advantage of obtaining correlation matrices from `sem` rather than from `correlate` is that you can perform statistical tests on the results, such as that the correlation of \( x1 \) and \( x3 \) is equal to the correlation of \( x2 \) and \( x3 \).

If you are willing to assume joint normality of the variables, you can obtain more efficient estimates of the correlations in the presence of missing-at-random data by specifying the `method(mlmv)` option.

Correlations are fit using `sem`.

See the following example:


**Higher-order CFA models**

Observed values sometimes measure traits or other aspects of latent variables, so we insert a new layer of latent variables to reflect those traits or aspects. We have measurements—say, \( x1, \ldots, x6 \)—all reflecting underlying factor \( F \), but \( x1 \) and \( x2 \) measure one trait of \( F \), \( x3 \) and \( x4 \) measure another trait, and \( x5 \) and \( x6 \) measure yet another trait. This model can be drawn as
The model can be written in command syntax as

\[(A\rightarrow x_1 \ x_2) \ (B\rightarrow x_3 \ x_4) \ (C\rightarrow x_5 \ x_6) \ (A \ B \ C\leftarrow F)\]

Higher-order CFA models can be fit using \texttt{sem} or \texttt{gsem}.

See the following example:

1. [SEM] Example 15. Higher-order CFA.

**Correlated uniqueness model**

Observed values sometimes are correlated just because of how the data are collected. Imagine we have factor T1 representing a trait with measurements x1 and x4. Perhaps T1 is aggression, and then x1 is self reported and x4 is reported by the spouse. Imagine we also have factor T2 with measurements x2 and x5. Again, x2 is self reported and x5 is reported by the spouse. It would not be unlikely that x1 and x2 are correlated and that x4 and x5 are correlated. That is exactly what the correlated uniqueness model assumes:
Data that exhibit this kind of pattern are known as multitrait–multimethod (MTMM) data. Researchers historically looked at the correlations, but structural equation modeling allows us to fit a model that incorporates the correlations.

The above model can be written in Stata command syntax as

```stata
(T1->x1 x4 x7) ///
(T2->x2 x5 x8) ///
(T3->x3 x6 x9), ///
cov(e.x1*e.x2 e.x1*e.x3 e.x2*e.x3) ///
cov(e.x4*e.x5 e.x4*e.x6 e.x5*e.x6) ///
cov(e.x7*e.x8 e.x7*e.x9 e.x8*e.x9)
```

An alternative way to type the above is to use the `covstructure()` option, which we can abbreviate as `covstruct()`:

```stata
(T1->x1 x4 x7) ///
(T2->x2 x5 x8) ///
(T3->x3 x6 x9), ///
covstruct(e.x1 e.x2 e.x3, unstructured) ///
covstruct(e.x4 e.x5 e.x6, unstructured) ///
covstruct(e.x7 e.x8 e.x9, unstructured)
```

Unstructured means that the listed variables have covariances. Specifying blocks of errors as unstructured would save typing if there were more variables in each block.

The correlated uniqueness model can be fit by `sem` or `gsem`, although we recommend use of `sem` in this case. Gaussian responses with the identity link are allowed to have correlated uniqueness (error) but only in the absence of censoring. `gsem` still provides the theoretical ability to fit these models in multilevel contexts, but convergence may be difficult to achieve.

See the following example:

Latent growth models

A latent growth model is a variation on the measurement model. In our measurement model examples, we have assumed four observed measurements of underlying factor $X$: $x_1$, $x_2$, $x_3$, and $x_4$. In the command language, we can write this as

$$(X \rightarrow x_1) (X \rightarrow x_2) (X \rightarrow x_3) (X \rightarrow x_4)$$

Let’s assume that the observed values are collected over time. $x_1$ is observed at time 0, $x_2$ at time 1, and so on. It thus may be more reasonable to assume that the observed values represent a base value and a growth modeled with a linear trend. Thus we might write the model as

$$(B@1 L@0 \rightarrow x_1) /// (B@1 L@1 \rightarrow x_2) /// (B@1 L@2 \rightarrow x_3) /// (B@1 L@3 \rightarrow x_4), /// noconstant$$

The equations for this model are

\[
\begin{align*}
x_1 &= B + 0L + e.x_1 \\
x_2 &= B + 1L + e.x_2 \\
x_3 &= B + 2L + e.x_3 \\
x_4 &= B + 3L + e.x_4
\end{align*}
\]

and the path diagram is

In evaluating this model, it is useful to review the means of the latent exogenous variables. In most models, latent exogenous variables have mean 0, and the means are thus uninteresting. `sem` usually constrains latent exogenous variables to have mean 0 and does not report that fact.

In this case, however, we ourselves have placed constraints, and thus the means are identified and in fact are an important point of the exercise. We must tell `sem` not to constrain the means of the two latent exogenous variables $B$ and $L$, which we do with the `means()` option:
We must similarly specify the `means()` option when using the Builder.

Latent growth models can be fit with `sem` or `gsem`.

See the following example:

1. **[SEM] Example 18.** Latent growth model.

## Models with reliability

A typical solution for dealing with variables measured with error is to find multiple measurements and use those measurements to develop a latent variable. See, for example, *Single-factor measurement models* and *Multiple-factor measurement models* above.

When the reliability of the variables is known—reliability is measured as the fraction of variances that is not due to measurement error—another approach is available. This approach can be used in place of or in addition to the use of multiple measurements. See **[SEM] sem and gsem option reliability()**.

Models with reliability can be fit with `sem` or `gsem`, although option `reliability()` is available only when responses are Gaussian with the identity link and only in the absence of censoring.

See the following example:

1. **[SEM] Example 24.** Reliability.

## Multilevel mixed-effects models

Multilevel modeling concerns the inclusion of common, random effects across groups of the data, which are known as levels. You have observations on students. Students attend schools. Or you have observations on patients. Patients are served by hospitals. In either case, there may be an unmeasured effect of the institution. Some schools are better than others while some schools are worse. The same applies to hospitals. These unmeasured effects can be parameterized as a latent variable that is constant within institution and varies across institution.

What we have just described is a two-level nested model. The first level, the lowest, is the observational level. The second level is school or hospital.

In a three-level nested model, students are served by schools which are served by counties, or patients are served by hospitals which are served by states. Whatever the example, there can be unmeasured effects of each of those higher levels contributing to the effect observed at the observational level.

An alternative to nested models is crossed models. People with jobs work in an industry and in a state. The same industries are found across states and the same states are found across industries, yet both may have an effect on some aspect of the lives of their workers.
Multilevel models are fit by \texttt{gsem}.

See the following examples:

1. \[\text{SEM}\] \textbf{Example 30g}. Two-level measurement model (multilevel, generalized response).
2. \[\text{SEM}\] \textbf{Example 38g}. Random-intercept and random-slope models (multilevel).
3. \[\text{SEM}\] \textbf{Example 39g}. Three-level model (multilevel, generalized response).
4. \[\text{SEM}\] \textbf{Example 40g}. Crossed models (multilevel).
5. \[\text{SEM}\] \textbf{Example 41g}. Two-level multinomial logistic regression (multilevel).
6. \[\text{SEM}\] \textbf{Example 42g}. One- and two-level mediation models (multilevel).

\begin{flushleft}
\textbf{Latent class models}
\end{flushleft}

A latent class model involves a latent variable that is categorical rather than continuous. The unobserved levels of the categorical latent variable are called latent classes. These classes correspond to unobserved groups in the population such as unobserved groups of healthy and unhealthy individuals or unobserved groups of consumers with different buying preferences. Latent class analysis can help us to identify and understand these groups.

When we fit a latent class model, we specify the number of latent classes and then estimate the probabilities of class membership. In addition, the parameters that are estimated in the rest of the model are allowed to vary across the classes. For example, we may fit intercept-only logistic regression models to a set of binary variables and allow the intercepts to be estimated separately across classes.

Latent class models are fit by \texttt{gsem}. The Stata command syntax for this type of model is

\begin{verbatim}
(y1 y2 y3 y4 <- ), logit lclass(C 2)
\end{verbatim}

This fits a latent class model with one categorical latent variable, \(C\), that has two classes. Both the name of the latent variable and the number of classes is specified in the \texttt{lclass()} option.

This basic latent class model can be extended in many ways.

1. We can specify that \(C\) has three, four, or even more latent classes.
2. We are not limited to having observed variables that are categorical. When variables are continuous, the analysis is sometimes called by other names such as latent profile analysis or latent cluster analysis instead of latent class analysis.
3. We can include predictors of \(C\)—predictors of the probabilities of being in the different classes.
4. We can include more than one categorical latent variable.

See the following examples:

1. \[\text{SEM}\] \textbf{Example 50g}. Latent class model.
2. \[\text{SEM}\] \textbf{Example 51g}. Latent class goodness-of-fit statistics.
3. \[\text{SEM}\] \textbf{Example 52g}. Latent profile model.
Finite mixture models

Finite mixture models also involve categorical latent variables. Here we focus on finite mixture regression models in which you can fit any regression model allowed by \texttt{gsem} and estimate the parameters of that model separately for each latent class. For a linear regression of \( y \) on \( x_1 \) and \( x_2 \), the command syntax for a two-class model is

\[
y \leftarrow x_1 \ x_2, \ lclass(C \ 2)
\]

The intercept and the coefficients on \( x_1 \) and \( x_2 \) will be estimated separately for the two classes. In addition, we estimate the probability of being in each class. If we have a variable \( z \) that predicts class membership, the command syntax becomes

\[
y \leftarrow x_1 \ x_2 \ (C \leftarrow z), \ lclass(C \ 2)
\]

The \texttt{fmm:} prefix can also be used to fit these finite mixture regression models. For instance, \texttt{fmm: regress} can fit the model shown above. \texttt{gsem} extends the types of models that can be fit using \texttt{fmm} by allowing more than one response variable or more than one categorical latent variable.

See the following examples:

1. [SEM] \textbf{Example 53g}. Finite mixture Poisson regression.
2. [SEM] \textbf{Example 54g}. Finite mixture Poisson regression, multiple responses.

References


Bauldry, S. 2014. \texttt{miivfind}: A command for identifying model-implied instrumental variables for structural equation models in \texttt{Stata}. \textit{Stata Journal} 14: 60–75.


Also see

[SEM] \textbf{Intro 4} — Substantive concepts

[SEM] \textbf{Intro 6} — Comparing groups

[SEM] \textbf{Example 1} — Single-factor measurement model
Description

*sem* and *gsem* have a unique feature that allows you to easily compare groups—compare males with females, compare age group 1 with age group 2 with age group 3, and so on—with respect to any SEM. Said more technically, any model fit by *sem* or *gsem* can be simultaneously estimated for different groups with some parameters constrained to be equal across groups and others allowed to vary, and those estimates can be used to perform statistical tests for comparing the groups.

*sem* and *gsem* have similar syntax for multiple group models, but there are important differences in default assumptions and in how you specify constraints. So we discuss how to compare groups using each command separately, first with *sem* and then with *gsem*.

Remarks and examples

Remarks are presented under the following headings:

Comparing groups with *sem*

- The generic SEM model
- *sem*: Fitting the model for different groups of the data
- *sem*: Which parameters vary by default, and which do not
- *sem*: Specifying which parameters are allowed to vary in broad, sweeping terms
- *sem*: Adding constraints for path coefficients across groups
- *sem*: Adding constraints for means, variances, or covariances across groups
- *sem*: Adding constraints for some groups but not others
- *sem*: Adding paths for some groups but not others
- *sem*: Relaxing constraints

Comparing groups with *gsem*

- *gsem*: Fitting the model for different groups of the data
- *gsem*: Which parameters vary by default, and which do not
- *gsem*: Specifying which parameters are allowed to vary in broad, sweeping terms
- *gsem*: Adding constraints for path coefficients across groups
- *gsem*: Adding constraints for means, variances, or covariances across groups
- *gsem*: Adding constraints for some groups but not others
- *gsem*: Adding paths for some groups but not others
- *gsem*: Relaxing constraints

Comparing groups with *sem*

*sem* and *gsem* have similar syntax for multiple group models. There are, however, important differences in default assumptions and in how you specify constraints, so we discuss how to compare groups using each command separately. We begin with *sem*’s syntax for group comparisons.
The generic SEM model

In [SEM] Intro 5, we noted that measurement models are often joined with other SEMs to produce

![Diagram of SEM model]

This can be written in the `sem` command syntax as

\[
(Y_1 \rightarrow \ldots) (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3}
\]
\[
(\ldots) \quad /// \quad \text{theoretical model stated in terms of}
\]
\[
(\ldots) \quad /// \quad \text{underlying concepts (latent variables)}
\]
\[
(\ldots) \quad ///
\]
\[
(X_1 \rightarrow \ldots) (X_2 \rightarrow \ldots)
\]

where the middle part is the theoretical model stated in terms of underlying concepts Y1, Y2, X1, and X2.

However we write the model, we are assuming the following:

1. The unobserved X1 and X2 are measured by the observed x1, x2, \ldots.
2. The middle part is stated in terms of the underlying concepts X1, X2, Y1, and Y2.
3. The unobserved Y1 and Y2 are measured by the observed y1, y2, \ldots.

**sem: Fitting the model for different groups of the data**

We can fit this model for different groups (say, age groups) by specifying the `group(varname)` option:

\[
(Y_1 \rightarrow \ldots) (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3}
\]
\[
(\ldots) \quad /// \quad \text{part 2}
\]
\[
(X_1 \rightarrow \ldots) (X_2 \rightarrow \ldots), \quad /// \quad \text{part 1}
\]
\[
\text{group(agegrp)}
\]
where \texttt{agegrp} is a variable in our dataset, perhaps taking on values 1, 2, 3, \ldots. We can specify the model by using the command language or by drawing the model in the Builder and then choosing and filling in the \texttt{group()} option.

After estimation, you can use \texttt{estat ginvariant} (see \texttt{SEM estat ginvariant}) to obtain Wald tests of whether constraints should be added and score tests of whether constraints should be relaxed.

\textbf{sem: Which parameters vary by default, and which do not}

When we specify \texttt{group(groupvar)}, the measurement parts of the model (parts 1 and 3) are constrained by default to be the same across the groups, whereas the middle part (part 2) will have separate parameters for each group. More specifically, parts 1 and 3 are constrained to be equal across groups except that the variances of the errors will be estimated separately for each group.

If there is no measurement component to the model—if there are no latent variables—then by default all parameters are estimated separately for each group.

\textbf{sem: Specifying which parameters are allowed to vary in broad, sweeping terms}

You can control which parameters are constrained to be equal across groups by specifying the \texttt{ginvariant()} option:

\[
(Y_1 \rightarrow ...) \quad (Y_2 \rightarrow ...), \quad /// \quad \text{part 1}
(X_1 \rightarrow ...) \quad (X_2 \rightarrow ...), \quad /// \quad \text{part 2}
(...), \quad /// \quad \text{part 3}
\]

\texttt{group(agegrp) ginvariant(pclasses)}

The parameter classes (\textit{pclasses}) for \texttt{sem} are as follows:

\begin{table}[h]
\centering
\begin{tabular}{|l|l|}
\hline
Parameter class description & Parameter class name \\
\hline
1. structural coefficients & scoef \\
2. structural intercepts & scons \\
3. measurement coefficients & mcoef \\
4. measurement intercepts & mcons \\
5. covariances of structural errors & serrvar \\
6. covariances of measurement errors & merrvar \\
7. covariances between structural and measurement errors & smerrcov \\
8. means of exogenous variables & meanex (*) \\
9. covariances of exogenous variables & covex (*) \\
10. all the above & all (*) \\
11. none of the above & none \\
\hline
\end{tabular}
\end{table}

(*): Be aware that 8, 9, and 10 (\textit{meanex}, \textit{covex}, and \textit{all}) exclude the observed exogenous variables—that is, they include only the latent exogenous variables—unless you specify the \texttt{noxconditional} option or the \texttt{noxconditional} option is otherwise implied; see \texttt{SEM sem option noxconditional}. This is what you would desire in most cases.
The default when `ginvariant()` is not specified is `ginvariant(mcoef mcons)`:

```
(Y1->...) (Y2->...) /// part 3, measurement
(...)
(...)
(X1->...) (X2->...), /// part 1, measurement
```

```
group(agegrp) ginvariant(mcoef mcons)
```

If you also wanted covariances of errors associated with measurement to be constrained across groups, you could type

```
(Y1->...) (Y2->...) /// part 3, measurement
(...)
(...)
(X1->...) (X2->...), /// part 1, measurement
```

```
group(agegrp) ginvariant(mcoef mcons merrvar)
```

**sem: Adding constraints for path coefficients across groups**

The `ginvariant()` option allows you to state in sweeping terms which parameters vary and which are invariant across groups. You can also constrain individual parameters to be equal across groups.

Pretend that in the substantive part of the generic model, we have `Y1<-Y2`. Assume that we fit the model and allow the structural part to vary across groups:

```
(Y1->...) (Y2->...) /// part 3, measurement
(...)
(Y1<-Y2) /// part 2, structural
(...)
(X1->...) (X2->...), /// part 1, measurement
```

```
group(agegrp)
```

In this model, the `Y1<-Y2` path coefficient is allowed to vary across groups by default. We could constrain the coefficient to be equal across groups by typing

```
(Y1->...) (Y2->...) /// part 3, measurement
(...)
(Y1<-Y2@b) /// part 2, structural
(...)
(X1->...) (X2->...), /// part 1, measurement
```

```
group(agegrp)
```

Note the `@b` in `(Y1<-Y2@b)` where we previously typed `Y1<-Y2`.

Constraining a coefficient to equal a symbolic name such as `b` is how we usually constrain equality, but in the usual case, the symbolic name appears at least twice in our model. For instance, we might have `(Y1<-Y2@b)` and `(Y1<-Y3@b)` and thus constrain path coefficients to be equal.

In the case above, however, `@b` appears only once. Because we specified `group(agegrp)`, results are as if we specified this model separately for each age group, and in each group, we are specifying `@b`. Thus we are constraining the path coefficient to be equal across all groups.
You use the same technique for adding constraints to means, variances, and covariances as you would for adding constraints to path coefficients. Remember that means are specified by the means() option, variances by the variance() option, and covariances by the covariance() option. The variance() and covariance() options are abbreviated var() and cov(), respectively.

You can specify, for instance,

$$(Y_1 \rightarrow \ldots) \ (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(Y_1 \leftarrow Y_2) \quad /// \quad \text{part 1, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(X_1 \rightarrow \ldots) \ (X_2 \rightarrow \ldots), \quad /// \quad \text{part 1, measurement}$$

$\text{group(agegrp)} \quad /// \quad \text{part 2, structural}$

$\text{means(X1@M)} \quad /// \quad \text{part 3, measurement}$

to constrain the mean of $X_1$ to be the same across groups. The means would have been different across groups by default.

You can specify

$$(Y_1 \rightarrow \ldots) \ (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(Y_1 \leftarrow Y_2) \quad /// \quad \text{part 1, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(X_1 \rightarrow \ldots) \ (X_2 \rightarrow \ldots), \quad /// \quad \text{part 1, measurement}$$

$\text{group(agegrp)} \quad /// \quad \text{part 2, structural}$

$\text{var(e.Y1@V)} \quad /// \quad \text{part 3, measurement}$

to constrain the variance of the error of $Y_1$ to be the same across groups.

If we wanted to allow the errors of $Y_1$ and $Y_2$ to be correlated (by default, errors are uncorrelated), we could add the cov() option:

$$(Y_1 \rightarrow \ldots) \ (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(X_1 \rightarrow \ldots) \ (X_2 \rightarrow \ldots), \quad /// \quad \text{part 1, measurement}$$

$\text{group(agegrp)} \quad /// \quad \text{part 2, structural}$

$\text{var(e.Y1@V)} \quad /// \quad \text{part 3, measurement}$

$\text{cov(e.Y1*e.Y2)} \quad /// \quad \text{part 2, structural}$

If we then wanted to constrain the covariance to be the same across groups, we would type

$$(Y_1 \rightarrow \ldots) \ (Y_2 \rightarrow \ldots) \quad /// \quad \text{part 3, measurement}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(\ldots) \quad /// \quad \text{part 2, structural}$$

$$(X_1 \rightarrow \ldots) \ (X_2 \rightarrow \ldots), \quad /// \quad \text{part 1, measurement}$$

$\text{group(agegrp)} \quad /// \quad \text{part 2, structural}$

$\text{var(e.Y1@V)} \quad /// \quad \text{part 3, measurement}$

$\text{cov(e.Y1*e.Y2@C)} \quad /// \quad \text{part 2, structural}$
sem: Adding constraints for some groups but not others

Consider the following model:

\[
... (Y1<-Y2) \ldots, \text{group(agegrp)}
\]

Above we saw how to constrain the \( Y1<-Y2 \) path coefficient to be the same across groups:

\[
... (Y1<-Y2@b) \ldots, \text{group(agegrp)}
\]

To constrain the path coefficients \( Y1<-Y2 \) to be equal for groups 1 and 2 but leave the \( Y1<-Y2 \) path coefficients unconstrained for the remaining groups, we would type

\[
... (Y1<-Y2) (1: Y1<-Y2@b) (2: Y1<-Y2@b) \ldots, \text{group(agegrp)}
\]

Think of this as follows:
1. \( (Y1<-Y2) \): We set a path for all the groups.
2. \( (1: Y1<-Y2@b) \): We modify the path for \( \text{agegrp} = 1 \).
3. \( (2: Y1<-Y2@b) \): We modify the path for \( \text{agegrp} = 2 \).
4. We do not modify the path for any other \( \text{agegrp} \) value.

The result is that we constrain age groups 1 and 2 to have the same value of the path, and we do not constrain the path for the other age groups.

You can constrain variance and covariance estimates to be the same across some groups but not others in the same way. You can specify, for instance,

\[
..., \text{group(agegrp)} \ var(1: e.Y1@V) \ var(2: e.Y1@V)
\]

or

\[
..., \text{group(agegrp)} \ cov(e.Y1*e.Y2) \ cov(1: e.Y1*e.Y2@C) \\
\hspace{1cm} \cov(2: e.Y1*e.Y2@C)
\]

Similarly, you can constrain means for some groups but not others, although this is rarely done:

\[
..., \text{group(agegrp)} \ means(1: X1@b) \ means(2: X1@b)
\]

sem: Adding paths for some groups but not others

In the same way that you can constrain coefficients for some groups but not others, you can add paths for some groups but not others. Consider the following model:

\[
... (Y1<-Y2) \ldots, \text{group(agegrp)}
\]

You can add the path \( Y1<-Y3 \) for groups 1 and 2 by typing

\[
... (Y1<-Y2) (1: Y1<-Y3) (2: Y1<-Y3) \ldots, \text{group(agegrp)}
\]

You can add covariances for some groups but not others in the same way. For instance, to allow the errors of \( Y1 \) and \( Y2 \) to be correlated in groups 1 and 2 only, you can specify

\[
..., \text{group(agegrp)} \ cov(1: e.Y1*e.Y2) \ cov(2: e.Y1*e.Y2)
\]
sem: Relaxing constraints

Just as you can specify

..., group(agegrp) ginvariant(pclasses)

and then add constraints, you can also specify

..., group(agegrp) ginvariant(pclasses)

and then relax constraints that the parameter classes impose.

For instance, if we specified ginvariant(scoef), then we would be constraining \((Y_1<-Y_2)\) to be invariant across groups. We could then relax that constraint by typing

\[
\text{... (Y1<-Y2) (1: Y1<-Y2@b1) (2: Y1<-Y2@b2) ...}, ///
group(agegrp) ginvariant(scoef)
\]

The path coefficients would be free in groups 1 and 2 and constrained in the remaining groups, if there are any. The path coefficient is free in group 1 because we specified symbolic name \(b_1\), and \(b_1\) appears nowhere else in the model. The path coefficient is free in group 2 because symbolic name \(b_2\) appears nowhere else in the model. If there are remaining groups and we want to relax the constraint on them, too, we would need to add \((3: Y1<-Y2@b3)\), and so on.

The same technique can be used to relax constraints on means, variances, and covariances:

\[
\text{... group(agegrp) ginvariant(... meanex ...)} ///
\text{means(1: X1@b1) means(2: X1@b2)}
\]

\[
\text{... group(agegrp) ginvariant(... serrvar ...)} ///
\text{var(1: e.Y1@V1) var(2: e.Y1@V2)}
\]

\[
\text{... group(agegrp) ginvariant(... serrvar ...)} ///
\text{cov(1: e.Y1*e.Y2@C1) cov(2: e.Y1*e.Y2@C2)}
\]

Comparing groups with gsem

With \text{gsem}, we fit multiple-group models in a similar way by using the \text{group()} and \text{ginvariant()} options. However, the allowed parameter classes and the default constraints differ between \text{sem} and \text{gsem}. Here, we will discuss how compare groups and how to set and release constraints across groups for models fit with \text{gsem}.

gsem: Fitting the model for different groups of the data

Let’s begin by considering a measurement model that has binary measurements of a latent variable. If we model the measurements using probit regression, we can fit a model with the command syntax as

\[(L \rightarrow y_1 y_2 y_3 y_4), \text{probit}\]

We can fit this model for different groups such as age groups by specifying the \text{group(varname)} option:

\[(L \rightarrow y_1 y_2 y_3 y_4), \text{probit group(agegrp)}\]
where `agegrp` is a variable in our dataset, perhaps taking on values 1, 2, 3, .... We can specify the model by using the command language or by drawing the model in the Builder and then choosing and filling in the `group()` option.

After estimation, you can use `test` to obtain Wald tests of whether constraints should be added. See [SEM] Example 49g for an example.

**gsem: Which parameters vary by default, and which do not**

When we specify `group(groupvar)` with `gsem`, the coefficients, intercepts, and cutpoints in the model are constrained by default to be the same across the groups. All other types of parameters will be estimated separately in each group. More specifically, all variances, covariances, means, and scale parameters will be estimated separately for each group.

**gsem: Specifying which parameters are allowed to vary in broad, sweeping terms**

You can control which parameters are constrained to be equal across groups by specifying the `ginvariant()` option:

```
(L -> y1 y2 y3 y4), probit ///
   group(agegrp) ginvariant(pclasses)
```

The parameter classes (`pclasses`) for `gsem` are as follows:

<table>
<thead>
<tr>
<th>Parameter class description</th>
<th>Parameter class name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. intercepts and cutpoints</td>
<td>cons</td>
</tr>
<tr>
<td>2. fixed coefficients</td>
<td>coef</td>
</tr>
<tr>
<td>3. latent variable coefficients</td>
<td>loading</td>
</tr>
<tr>
<td>4. covariances of errors</td>
<td>errvar</td>
</tr>
<tr>
<td>5. scaling parameters</td>
<td>scale</td>
</tr>
<tr>
<td>6. means of exogenous variables</td>
<td>means</td>
</tr>
<tr>
<td>7. covariances of exogenous latent variables</td>
<td>covex</td>
</tr>
<tr>
<td>8. all the above</td>
<td>all</td>
</tr>
<tr>
<td>9. none of the above</td>
<td>none</td>
</tr>
</tbody>
</table>

The default when `ginvariant()` is not specified is `ginvariant(cons coef loading)`:  

```
(L -> y1 y2 y3 y4), probit ///
   group(agegrp) ginvariant(cons coef loading)
```

If you also wanted the variance of the latent variable to be constrained across groups, you could type

```
(L -> y1 y2 y3 y4), probit ///
   group(agegrp) ginvariant(cons coef loading covex)
```

because `L` is an exogenous variable and `covex` refers to all covariances and variances of exogenous variables.
gsem: Adding constraints for path coefficients across groups

The ginvariant() option allows you to state in sweeping terms which parameters vary and which are invariant across groups. You can also constrain individual parameters to be equal across groups.

If we fit a model by typing

\[(L \rightarrow y1 \ y2 \ y3 \ y4), \text{probit} /// \]
\[\text{group(agegrp) ginvariant(cons)}\]

In this model, only the intercepts are constrained to be equal across groups. The \(L\rightarrow y1, L\rightarrow y2, L\rightarrow y3,\) and \(L\rightarrow y4\) path coefficients are allowed to vary across groups because we did not include loading in the ginvariant() option. We could constrain the \(L\rightarrow y4\) coefficient to be equal across groups by typing

\[(L \rightarrow y1 \ y2 \ y3 \ y4@b), \text{probit} /// \]
\[\text{group(agegrp) ginvariant(cons)}\]

Note the @b attached to \(y4\).

Constraining a coefficient to equal a symbolic name such as \(b\) is how we usually constrain equality, but in the usual case, the symbolic name appears at least twice in our model. For instance, we might have \((L\rightarrow y2@b)\) and \((L\rightarrow y3@b)\) and thus constrain path coefficients to be equal.

In the case above, however, @b appears only once. Because we specified group(agegrp), results are as if we specified this model separately for each age group, and in each group, we are specifying @b. Thus we are constraining the path coefficient to be equal across all groups.

gsem: Adding constraints for means, variances, or covariances across groups

You use the same technique for adding constraints to means, variances, and covariances as you would for adding constraints to path coefficients. Remember that means are specified by the means() option, variances by the variance() option, and covariances by the covariance() option. The variance() and covariance() options are abbreviated var() and cov(), respectively.

You can specify, for instance,

\[(L \rightarrow y1 \ y2 \ y3 \ y4), \text{probit} /// \]
\[\text{group(agegrp) ginvariant(cons)} \text{var(L@v)}\]

to constrain the variance of \(L\) to be the same across groups.

gsem: Adding constraints for some groups but not others

Again, we consider the following model:

\[(L \rightarrow y1 \ y2 \ y3 \ y4), \text{probit} /// \]
\[\text{group(agegrp) ginvariant(cons)}\]

Above we saw how to constrain the \(L\rightarrow y4\) path coefficient to be the same across groups:

\[(L \rightarrow y1 \ y2 \ y3 \ y4@b), \text{probit} /// \]
\[\text{group(agegrp) ginvariant(cons)}\]
To constrain the path coefficients $L \rightarrow y_4$ to be equal for groups 1 and 2 but leave the $L \rightarrow y_4$ path coefficients unconstrained for the remaining groups, we would type

$$(L \rightarrow y_1 y_2 y_3 y_4) \ (1: L \rightarrow y_4@b) \ (2: L \rightarrow y_4@b), \probit \ /// \ 
group(agegrp) \ ginvariant(cons)$$

Think of this as follows:

1. $(L \rightarrow y_4)$: We set a path for all the groups.
2. $(1: L \rightarrow y_4@b)$: We modify the path for $agegrp = 1$.
3. $(2: L \rightarrow y_4@b)$: We modify the path for $agegrp = 2$.
4. We do not modify the path for any other $agegrp$ value.

The result is that we constrain age groups 1 and 2 to have the same value of the path, and we do not constrain the path for the other age groups.

You can constrain variance, covariance, and mean estimates to be the same across some groups but not others in the same way. You can specify, for instance,

$$(L \rightarrow y_1 y_2 y_3 y_4), \probit \ /// \ 
group(agegrp) \ var(1: L\@v) \ var(2: L\@v)$$

gsem: Adding paths for some groups but not others

In the same way that you can constrain coefficients for some groups but not others, you can add paths for some groups but not others. Consider the following model:

$$(L \rightarrow y_1 y_2 y_3), \probit \ group(agegrp)$$

You can add the path $L \rightarrow y_4$ for groups 1 and 2 by typing

$$(L \rightarrow y_1 y_2 y_3) \ (1: L \rightarrow y_4) \ (2: L \rightarrow y_4), \probit \ group(agegrp)$$

gsem: Relaxing constraints

Just as you can specify

$$..., \ group(agegrp) \ ginvariant(pc\_classes)$$

and then add constraints, you can also specify

$$..., \ group(agegrp) \ ginvariant(pc\_classes)$$

and then relax constraints that the parameter classes impose.

For instance, if we specified $ginvariant(loading)$, then we would be constraining $(L \rightarrow y_4)$ to be invariant across groups. We could then relax that constraint by typing

$$(L \rightarrow y_1 y_2 y_3 y_4) \ (1: L \rightarrow y_4@b1) \ (2: L \rightarrow y_4@b2), \probit \ /// \ 
group(agegrp) \ ginvariant(... \ loading)$$

The $(L \rightarrow y_4)$ path coefficients would be free in groups 1 and 2 and constrained in the remaining groups, if there are any. The path coefficient is free in group 1 because we specified symbolic name $b_1$, and $b_1$ appears nowhere else in the model. The path coefficient is free in group 2 because symbolic name $b_2$ appears nowhere else in the model. If there are remaining groups and we want to relax the constraint on them, too, we would need to add $(3: L \rightarrow y_4@b3)$, and so on.
The same technique can be used to relax constraints on means, variances, and covariances. For instance,

```
(L -> y1 y2 y3 y4), probit ///
group(agegrp) ginvariant(... covex) var(1: L@V1) var(2: L@V2)
```

Reference


Also see

[SEM] Intro 5 — Tour of models
[SEM] Intro 7 — Postestimation tests and predictions
[SEM] sem and gsem option covstructure() — Specifying covariance restrictions
[SEM] sem group options — Fitting models on different groups
Description

After fitting a model with \texttt{sem} or \texttt{gsem}, you can perform statistical tests, obtain predicted values, and more. Everything you can do is listed below.

\texttt{sem} and \texttt{gsem} vary in the tests and features available after estimation, and we mark whether each test and feature is available after \texttt{sem}, \texttt{gsem}, or both.

Remarks and examples

Remarks are presented under the following headings:

- Replaying the model (\texttt{sem} and \texttt{gsem})
- Displaying odds ratios, incidence-rate ratios, etc. (\texttt{gsem} only)
- Obtaining goodness-of-fit statistics (\texttt{sem} and \texttt{gsem})
- Performing tests for including omitted paths and relaxing constraints (\texttt{sem} only)
- Performing tests of model simplification (\texttt{sem} and \texttt{gsem})
- Displaying other results, statistics, and tests (\texttt{sem} and \texttt{gsem})
- Obtaining predicted values (\texttt{sem})
- Obtaining predicted values (\texttt{gsem})
- Using contrast, \texttt{pwcompare}, and margins (\texttt{sem} and \texttt{gsem})
- Accessing stored results

Replaying the model (\texttt{sem} and \texttt{gsem})

After estimation, you can type \texttt{sem} or \texttt{gsem} without arguments to display the estimation output:

\begin{verbatim}
. sem
(original output reappears)
\end{verbatim}

If you wish to see results in the Bentler–Weeks formulation, after \texttt{sem} estimation type

\begin{verbatim}
. estat framework
(output omitted)
\end{verbatim}

See \cite{SEM} Example 11.

In many of the postestimation commands listed below for use after \texttt{sem} and \texttt{gsem}, you will need to refer symbolically to particular coefficients. For instance, in the model

\begin{verbatim}
. sem ... (Y<-x1) ... , ... cov(e.Y1*e.Y2)
\end{verbatim}

the symbolic name of the coefficient corresponding to the path \(Y<-x1\) is \texttt{\_b[Y1:x1]}, and the symbolic name of the coefficient corresponding to the covariance of \(e.Y1\) and \(e.Y2\) is \texttt{\_b[/cov(e.Y1,e.Y2)]}.

Figuring out what the names are can be difficult, so instead, type

\begin{verbatim}
. sem, coeflegend
\end{verbatim}

or

\begin{verbatim}
. gsem, coeflegend
\end{verbatim}

With this command, \texttt{sem} (\texttt{gsem}) will produce a table looking very much like the estimation output that lists the \texttt{\_b[\_]} notation for the estimated parameters in the model; see \cite{SEM} Example 8.
Displaying odds ratios, incidence-rate ratios, etc. (gsem only)

In some generalized linear response functions, exponentiated coefficients have a special meaning. Those special meanings are

<table>
<thead>
<tr>
<th>Common name</th>
<th>Family</th>
<th>Link</th>
<th>Meaning of exp(coef)</th>
</tr>
</thead>
<tbody>
<tr>
<td>logit</td>
<td>Bernoulli</td>
<td>logit</td>
<td>odds ratio</td>
</tr>
<tr>
<td>ologit</td>
<td>ordinal</td>
<td>logit</td>
<td>odds ratio</td>
</tr>
<tr>
<td>mlogit</td>
<td>multinomial</td>
<td>logit</td>
<td>relative-risk ratio</td>
</tr>
<tr>
<td>Poisson</td>
<td>Poisson</td>
<td>log</td>
<td>incidence-rate ratio</td>
</tr>
<tr>
<td>nbreg</td>
<td>nbreg</td>
<td>log</td>
<td>incidence-rate ratio</td>
</tr>
</tbody>
</table>

For survival models, the special meanings of the exponentiated coefficients are

<table>
<thead>
<tr>
<th>Survival distribution</th>
<th>Meaning of exp(coef)</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential</td>
<td>hazard ratio</td>
</tr>
<tr>
<td>Weibull</td>
<td>hazard ratio</td>
</tr>
<tr>
<td>gamma</td>
<td>time ratio</td>
</tr>
<tr>
<td>loglogistic</td>
<td>time ratio</td>
</tr>
<tr>
<td>lognormal</td>
<td>time ratio</td>
</tr>
</tbody>
</table>

\texttt{gsem} reports coefficients, not exponentiated coefficients. You can obtain exponentiated coefficients and their standard errors by using \texttt{estat eform} after estimation. Using \texttt{estat eform} is no different from redisplaying results. The syntax is

\texttt{estat eform equationname}

After \texttt{gsem}, equations are named after the dependent variable, so if you want to see the equation for cases in exponentiated form, you can type \texttt{estat eform cases}.

See \texttt{SEM Example 33g}, \texttt{SEM Example 34g}, \texttt{SEM Example 47g}, \texttt{SEM Example 48g}, and \texttt{SEM estat eform}.

Obtaining goodness-of-fit statistics (sem and gsem)

One goodness-of-fit statistic and test is reported at the bottom of the \texttt{sem} output:

\texttt{. sem // redisplay results}

\texttt{Variables in structural equation model (output omitted)}

\texttt{Structural equation model (coefficient table omitted)}

\texttt{LR test of model vs. saturated: chi2(2) = 1.78, Prob > chi2 = 0.4111}

Be warned that this test is based on the assumption of joint normality of the observed variables. In the case of nonnormal data, you may specify the \texttt{vce(sbentler)} option to obtain the Satorra–Bentler (1994) scaled $\chi^2$ test. In either case, the test is a goodness-of-fit test in badness-of-fit units; a significant result implies that the model does not fit well. More mathematically, the null hypothesis of this test is that the fitted covariance matrix and mean vector of the observed variables are equal to the matrix and vector observed in the population as measured by the sample. Remember, however, the goal is not to maximize the goodness of fit. One must not add paths or covariances that are not theoretically meaningful.
In addition, other goodness-of-fit statistics are available:

1. (sem only.) Command `estat gof` reports a variety of goodness-of-fit statistics; see [SEM] `estat gof` and [SEM] Example 4.

2. (sem only.) Command `estat eqgof` reports $R^2$-like goodness-of-fit statistics for each equation separately; see [SEM] `estat eqgof` and [SEM] Example 3.

3. (sem only.) Command `estat ggof` reports goodness-of-fit statistics by group when you have estimated using `sem`’s `group()` option; see [SEM] `estat ggof` and [SEM] Example 21.

4. (sem only.) Command `estat residuals` reports the element-by-element differences between the observed and fitted covariance matrix, and the observed and fitted mean vector, optionally in standardized or in normalized units; see [SEM] `estat residuals` and [SEM] Example 10.

5. (sem and gsem.) Commands `estat ic` and `estimates stats` report the Akaike and Bayesian information criteria; see [SEM] Example 51g, [SEM] Example 52g, [R] `estat ic`, and [R] `estimates stats`.

6. (gsem only.) Command `estat lcgof` reports goodness-of-fit statistics for latent class models; see [SEM] `estat lcgof` and [SEM] Example 51g.

Performing tests for including omitted paths and relaxing constraints (sem only)

1. (sem only.) Command `estat mindices` reports $\chi^2$ modification indices and significance values for each omitted path in the model, along with the expected parameter change; see [SEM] `estat mindices`, [SEM] Example 5, and [SEM] Example 9.

2. (sem only.) Command `estat scoretests` performs score tests on each of the linear constraints placed on the paths and covariances; see [SEM] `estat scoretests` and [SEM] Example 8.

3. (sem only.) Command `estat ginvariant` is for use when you have estimated using `sem`’s `group()` option; see [SEM] Intro 6. This command tests whether you can relax constraints that parameters are equal across groups; see [SEM] `estat ginvariant` and [SEM] Example 22.

Performing tests of model simplification (sem and gsem)

1. (sem and gsem.) Command `test` reports Wald tests of single or multiple linear constraints. See [SEM] Example 8 and [SEM] `test`.

2. (sem and gsem.) Command `lrtest` reports likelihood-ratio tests of single or multiple linear constraints. See [SEM] Example 10, [SEM] `lrtest`, and [SEM] Example 39g.

3. (sem only.) Command `estat eqtest` reports an overall Wald test for each equation in the model, the test corresponding to all coefficients in the equation except the intercept being simultaneously 0; see [SEM] `estat eqtest` and [SEM] Example 13.

4. (sem only.) Command `estat ginvariant` is for use when you have estimated using `sem`’s `group()` option; see [SEM] Intro 6. This command tests whether parameters allowed to vary across groups could be constrained; see [SEM] `estat ginvariant` and [SEM] Example 22.
Displaying other results, statistics, and tests (sem and gsem)

1. (sem only.) The estat stdize: command prefix—used in front of test, testnl, lincom, and nlcom—allows you to perform tests on standardized coefficients. See [SEM] estat stdize and [SEM] Example 16.

2. (sem only.) Command estat teffects reports total effects of one variable on another and decomposes the total effect into direct and indirect effects. Results can be reported in standardized or unstandardized form. See [SEM] estat teffects and [SEM] Example 7.

3. (sem only.) Command estat stable assesses the stability of nonrecursive structural equation systems; see [SEM] estat stable and [SEM] Example 7.

4. (sem and gsem.) Command estat summarize reports summary statistics for the observed variables used in the model; see [SEM] estat summarize.

5. (sem and gsem.) Command lincom reports the value, standard error, significance, and confidence interval for linear combinations of estimated parameters; see [SEM] lincom.

6. (sem and gsem.) Command nlcom reports the value, standard error, significance, and confidence interval for nonlinear (and linear) combinations of estimated parameters; see [SEM] nlcom and [SEM] Example 42g.

7. (sem and gsem.) Command estat vce reports the variance–covariance matrix of the estimated parameters; see [R] estat vce.

8. (gsem only.) Command estat lcmean reports marginal predicted means of each outcome within each latent classes; see [SEM] estat lcmean, [SEM] Example 50g, and [SEM] Example 53g.

9. (gsem only.) Command estat lcprob reports marginal predicted latent class probabilities; see [SEM] estat lcprob, [SEM] Example 50g, and [SEM] Example 53g.

Obtaining predicted values (sem)

You obtain predicted values with the predict command. Below we will write that predictions are the expected values, but be aware that when there are latent variables in your model, predictions are based on predicted scores; the scores can be inconsistent, and thus any prediction based on them can be inconsistent.

Available are the following:

1. predict newvar, xb(odepvarname) creates new variable newvar containing the predicted values for observed endogenous variable odepvarname.

   predict stub*, xb creates new variables stub1, stub2, ... containing the predicted values for all the observed endogenous variables in the model.

   These predicted values are the expected value of the xb given the values of the observed exogenous variables.

2. predict newvar, latent(Lname) creates new variable newvar containing the predicted values of the latent variable Lname, whether endogenous or exogenous.

   predict stub*, latent creates new variables stub1, stub2, ... containing the predicted values for all the latent variables in the model.

   Predicted values of latent variables, also known as predicted factor scores, are the expected values of the variables given the values of the observed variables.
3. predict newvar, xblatent(Lname) creates new variable newvar containing the predicted values for latent endogenous variable Lname.

predict stub*, xblatent creates new variables stub1, stub2, ... containing the predicted values for all the latent endogenous variables in the model.

predict with xblatent(Lname) differs from latent(Lname) in that the factor scores predicted by latent() are then used with the linear equation for Lname to make the prediction.

4. predict stub*, scores will create a slew of variables, one for each estimated parameter, containing the observation-by-observation values of the first derivative, also known as scores. This command is intended for use by programmers and may only be used after estimation using method(ml) or method(mlmv).

See [SEM] Example 14 and [SEM] predict after sem.

Obtaining predicted values (gsem)

predict after gsem is more complicated than predict after sem because generalized SEMs are more complicated. Three new issues arise: the prediction of generalized linear response variables as opposed to linear response variables, the optional presence of multilevel continuous latent variables in the model, and the optional presence of categorical latent variables in the model.

Let’s start with the response functions of the observed endogenous variables. Corresponding to predict, xb after sem are the following:

1. predict newvar, mu outcome(depvar) predicts mean responses in natural units—linear, probability, counts, and so on. Think $g^{-1}(x_i b)$ in the generalized linear model notation.

   With continuous latent variables, calculations are made using the empirical Bayes mean predictions of latent variables, which can be computationally intensive. You can specify that empirical Bayes modes be used by specifying option conditional(ebmodes). You can also specify that all continuous latent variables be set to their mean value, typically 0, by specifying the conditional(fixedonly) option.

   With categorical latent variables, calculations can be made for a specific latent class by specifying the class(lcspec) option.

2. predict newvar, mu outcome(depvar) marginal also predicts mean responses in natural units, and again it is appropriate to think $g^{-1}(x_i b)$.

   With continuous latent variables, calculations are made by integrating the prediction formula with respect to the continuous latent variables, which can be computationally intensive.

   With categorical latent variables, the overall mean is computed by a weighted sum of the latent class means, where the weights are latent class probabilities.

3. predict newvar, mu outcome(depvar) pmarginal also predicts mean responses in natural units, and again it is appropriate to think $g^{-1}(x_i b)$.

   The pmarginal option is allowed only for models with categorical latent variables. The overall mean is computed by a weighted sum of the latent class means, where the weights are posterior latent class probabilities.

4. Option pr is a synonym for mu when using family-and-link combinations that produce probabilities, such as logit, probit, etc.
5. predict newvar, eta outcome(depvar) calculates the linear prediction $x_i b$.

With continuous latent variables, calculations are made using the empirical Bayes mean predictions of latent variables, which can be computationally intensive. You can specify that empirical Bayes modes be used by specifying option conditional(ebmodes). You can also specify that all continuous latent variables be set to their mean value, typically 0, by specifying option conditional(fixedonly).

With categorical latent variables, calculations can be made for a specific latent class by specifying the class(lcspec) option.

6. predict newvar, density outcome(depvar) calculates the density function for depvar using the current data and model parameters.

With continuous latent variables, calculations are made using the empirical Bayes mean predictions of latent variables, which can be computationally intensive. You can change the treatment of latent variables by specifying option conditional(ebmodes), conditional(fixedonly), or marginal.

With categorical latent variables, calculations can be made for a specific latent class by specifying the class(lcspec) option.

7. predict newvar, distribution outcome(depvar) calculates the cumulative distribution function for depvar using the current data and model parameters.

With continuous latent variables, calculations are made using the empirical Bayes mean predictions of latent variables, which can be computationally intensive. You can change the treatment of latent variables by specifying option conditional(ebmodes), conditional(fixedonly), or marginal.

With categorical latent variables, calculations can be made for a specific latent class by specifying the class(lcspec) option.

Option distribution is not allowed with family(multinomial).

8. predict newvar, survival outcome(depvar) calculates the cumulative survivor function for depvar using the current data and model parameters.

With continuous latent variables, calculations are made using the empirical Bayes mean predictions of latent variables, which can be computationally intensive. You can change the treatment of latent variables by specifying option conditional(ebmodes), conditional(fixedonly), or marginal.

With categorical latent variables, calculations can be made for a specific latent class by specifying the class(lcspec) option.

Option survival is allowed only with family(exponential), family(gamma), family(loglogistic), family(lognormal), and family(weibull).

9. Option outcome() varies to accommodate the multiple predictions produced by multinomial logit, ordinal probit, ordinal logit, and ordinal cloglog.

   a. outcome(depvar #) specifies that you want the prediction for depvar = #.

   b. outcome(#.depvar) is another way of specifying that you want the prediction for depvar = #.

   c. outcome(depvar ##) specifies that you want the prediction for the #th category.

For multinomial logit, you use the above form of outcome() when requesting mu or eta predictions.
For ordered outcomes, you use the above form of `outcome()` when requesting \( \mu \) predictions. For \( \eta \) predictions, there is only one linear equation, and you use the usual `outcome(depvar)` form.

10. Option `class(lclspec)` is for models with categorical latent variables and allows you to target predictions for a specific latent class. For models with one categorical latent variable, `lclspec` can be a level value, such as `class(2)` or its equivalent factor-variable notation `class(2.C)`, assuming the latent class variable is \( C \). For models with two or more categorical latent variables, `lclspec` may only be in factor-variable notation, such as `class(2.C#1.D)` for categorical latent variables \( C \) and \( D \).

11. Some GLM families, such as Poisson, allow specification of an offset or exposure. If you use one of those families and you specify an offset or exposure with `gsem` when you fit the model and now want predictions with offset \( = 0 \) (exposure \( = 1 \)), you can specify option `nooffset`.

12. To obtain predicted values for all observed endogenous variables, you can omit the `outcome()` option and replace `newvar` with `stub*`—that is, a piece of a new variable name followed by `*`. This is not just convenient; it can also save computer time because some parts of the calculation can be shared.

For example, you can change

```
predict newvar, mu outcome(depvar)
```

to be

```
predict new*, mu
```

So much for the observed endogenous variables. Let’s now move on to continuous latent variables, whether endogenous or exogenous:

13. `predict newvar, latent(Lname)` predicts the value of the specified latent variable by using empirical Bayes means. You can specify that empirical Bayes modes be used by specifying option `ebmodes`; see item 16b below.

14. `predict newvar, latent(Lname) se(another_newvar)` will predict both the latent variable and its standard error.

15. You can use the same `stub*` trick mentioned for predicting observed endogenous variables to obtain all the latent variables. Replace `newvar` with `stub*` and omit `Lname` to produce

```
predict stub*, latent
```

You can do this with the `se()` option, too:

```
predict stub*, latent se(another.stub*)
```

16. Four options are related to the iterative procedure for obtaining the predicted values:

   a. `ebmodes` specifies that empirical Bayes modes rather than means be predicted. Because latent variables are assumed to be normally distributed, means and modes are usually similar. They are, however, calculated differently. The predictions are the means or modes of the empirical posterior distributions of the latent variables, which are not necessarily normal. Therefore, the means can differ from the modes. Both are computationally intensive.

   The default empirical Bayes means calculation uses adaptive quadrature, which is to say, numerical integration. All three of the options listed below apply to this calculation.
The alternative empirical Bayes mode calculation uses an optimization method that does not require adaptive quadrature. For small datasets, modes can usually be calculated more quickly than means. For large datasets, however, predicting modes can actually take longer. If you specify `ebmodes`, of the three options listed below, only `tolerance()` and `iterate()` are relevant.

b. `intpoints(#)` specifies the number of quadrature (numerical integration) points to be used to compute the empirical Bayes means. The default is the value used in estimation, which in turn defaults to 7.

This option also controls the number of quadrature points used to compute marginal predictions for the observed endogenous variables.

c. `tolerance(#)` specifies the convergence tolerance in computing empirical Bayes means and modes and defaults to the value used in estimation, which in turn defaults to 1.0e–8.

d. `iterate(#)` specifies the maximum number of iterations in computing empirical Bayes means and modes and defaults to the value used in estimation, which in turn defaults to 1,001.

Continuous latent variables come in observation-level and multilevel flavors. If you have a multilevel latent variable such as `M1[school]`, you can predict it just as you would any other latent variable:

```
predict newvar, latent(M1[school])
```

If you predict all the latent variables with the `stub*` trick, any multilevel latent variables will be automatically included among the new variables.

The remaining predictions are for models with categorical latent variables.

17. `predict newvar, classpr class(lcspec)` predicts the probability of belonging to the specified latent class. This probability is the same one used to combine the class specific densities to form the marginal likelihood.

18. `predict newvar, classposteriorpr class(lcspec)` predicts the posterior probability of belonging to the specified latent class. This probability is a function of the above latent class computed by `predict, classpr` and the fitted outcome densities.

19. You can use the same `stub*` trick mentioned above to obtain predicted probabilities for all latent classes. Replace `newvar` with `stub*` and omit `class()` to produce

```
predict stub*, classpr
```

or

```
predict stub*, classposteriorpr
```

See [SEM] Example 28g, [SEM] Example 29g, [SEM] Example 50g, [SEM] Example 52g, and [SEM] predict after gsem.

Using contrast, pwcompare, and margins (sem and gsem)

`contrast`, `pwcompare`, and `margins` are postestimation commands of Stata. All three can be used after `gsem`, and `margins` can be used after `sem` as well. Even so, these commands work best when you specify your model with Stata’s factor-variable notation; see [SEM] Intro 3. `sem` does not allow that notation, and so obtaining desired results is a little more work.

1. Command `contrast` can be used after `gsem` to test linear hypotheses and make comparisons involving factor variables and their interactions. `contrast` allows for performing ANOVA-style tests of main effects, simple effects, or interaction effects. In addition, effects can be
decomposed into specific types of contrasts such as comparison against the grand mean, orthogonal polynomial contrasts, and more. See [R] contrast.

2. Command \texttt{pwcompare} can be used after \texttt{gsem} to obtain all pairwise comparisons of marginal linear predictions, including cell means, marginal means, etc. See [R] \texttt{pwcompare}.

3. Command \texttt{margins} is used after \texttt{sem} or \texttt{gsem} to obtain margins, meaning estimated marginal means, predictive margins, adjusted predictions, and marginal effects. \texttt{margins} also allows for computing contrasts and pairwise comparisons of both linear and nonlinear predictions. See [SEM] \texttt{Example 50g}, [R] \texttt{margins}, [R] \texttt{margins}, \texttt{contrast}, and [R] \texttt{margins}, \texttt{pwcompare}.

\section*{Accessing stored results}

\texttt{sem} and \texttt{gsem} store all results in \texttt{e()}; see \emph{Stored results} in [SEM] \texttt{sem} and \emph{Stored results} in [SEM] \texttt{gsem}. To get some idea of what is stored in \texttt{e()} after \texttt{sem} or \texttt{gsem} estimation, type

\begin{verbatim}
. ereturn list
(output omitted)
\end{verbatim}

You can save estimation results in files or temporarily in memory and do other useful things with them; see [R] \texttt{estimates}.

Not stored by \texttt{sem} in \texttt{e()} are the Bentler–Weeks matrices, but they can be obtained from the \texttt{r()} stored results of \texttt{estat framework}. (The Bentler–Weeks matrices are not relevant in the case of \texttt{gsem}.)

See [SEM] \texttt{sem} and [SEM] \texttt{estat framework}.

\section*{Reference}


\section*{Also see}

[SEM] \texttt{Intro 6} — Comparing groups

[SEM] \texttt{Intro 8} — Robust and clustered standard errors
**Intro 8 — Robust and clustered standard errors**

**Description**

`sem` and `gsem` provide two options to modify how standard-error calculations are made: `vce(robust)` and `vce(cluster clustvar)`. These standard errors are less efficient than the default standard errors, but they are valid under less restrictive assumptions.

These options are allowed only when default estimation method `method(ml)` is used or when option `method(mlmv)` is used. `ml` stands for maximum likelihood, and `mlmv` stands for maximum likelihood with missing values; see [SEM] Intro 4, [SEM] sem, and [SEM] gsem.

Also see [SEM] Intro 9, entitled Standard errors, the full story.

**Options**

`vce(vcetype)` specifies how the VCE, and thus the standard errors, is calculated. VCE stands for variance–covariance matrix of the estimators. The standard errors that `sem` and `gsem` report are the square roots of the diagonal elements of the VCE.

`vce(oim)` is the default. `oim` stands for observed information matrix (OIM). The information matrix is the matrix of second derivatives, usually of the log-likelihood function. The OIM estimator of the VCE is based on asymptotic maximum-likelihood theory. The VCE obtained in this way is valid if the errors are independent and identically distributed normal, although the estimated VCE is known to be reasonably robust to violations of the normality assumption, at least as long as the distribution is symmetric and normal-like.

`vce(robust)` specifies an alternative calculation for the VCE, called robust because the VCE calculated in this way is valid under relaxed assumptions. The method is formally known as the Huber/White/sandwich estimator. The VCE obtained in this way is valid if the errors are independent and identically distributed normal, although the estimated VCE is known to be reasonably robust to violations of the normality assumption, at least as long as the distribution is symmetric and normal-like.

`vce(cluster clustvar)` is a generalization of the `vce(robust)` calculation that relaxes the assumption of independence of the errors and replaces it with the assumption of independence between clusters. Thus the errors are allowed to be correlated within clusters.

**Remarks and examples**

The `vce()` option is allowed by `sem` and `gsem`. In the rest of this entry, we will use `sem` in illustrations, but everything we say applies equally to `gsem`.

The `vce(robust)` option,

```
    . sem ..., ... vce(robust)
```

and the `vce(cluster clustvar)` option,

```
    . sem ..., ... vce(cluster clustvar)
```
relax assumptions that are sometimes unreasonable for a given dataset and thus produce more accurate standard errors in those cases. Those assumptions are homoskedasticity of the variances of the errors—vce(robust)—and independence of the observations—vce(cluster clustvar). vce(cluster clustvar) relaxes both assumptions.

Homoskedasticity means that the variances of the errors are the same from observation to observation. Homoskedasticity can be unreasonable if, for instance, the error corresponds to a dependent variable of income or socioeconomic status. It would not be unreasonable to instead assume that, in the data, the variance of income or socioeconomic status increases as the mean increases. In such cases, rather than typing

```
. sem (y<-...) (...) (...<-x1) (...<-x2)
```

you would type

```
. sem (y<-...) (...) (...<-x1) (...<-x2), vce(robust)
```

Independence implies that the observations are uncorrelated. If you have observations on people, some of whom live in the same neighborhoods, it would not be unreasonable to assume instead that the error of one person is correlated with those of others who live in the same neighborhood because neighborhoods tend to be homogeneous. In such cases, if you knew the neighborhood, rather than typing

```
. sem (y<-...) (...) (...<-x1) (...<-x2)
```

you would type

```
. sem (y<-...) (...) (...<-x1) (...<-x2), vce(cluster neighborhood)
```

Understand that if the assumptions of independent and identically distributed normal errors are met, the vce(robust) and vce(cluster clustvar) standard errors are less efficient than the standard vce(oim) standard errors. Less efficient means that for a given sample size, the standard errors jump around more from sample to sample than would the vce(oim) standard errors. vce(oim) standard errors are unambiguously best when the standard assumptions of homoskedasticity and independence are met.

Also see

[SEM] Intro 7 — Postestimation tests and predictions
[SEM] Intro 9 — Standard errors, the full story
[SEM] gsem estimation options — Options affecting estimation
[SEM] sem option method() — Specifying method and calculation of VCE
In [SEM] Intro 8, we told you part of the story of the calculation of the VCE, the part we wanted to emphasize. In this section, we tell you the full story.

We at Stata try to draw a clear distinction between method and technique. The method is the process used to obtain the parameter estimates. The technique is the process used to obtain the variance–covariance matrix of the parameter estimates, which is to say, the standard errors.

The literature does not always draw such clear distinctions.

*sem* and *gsem* provide the following methods and techniques:

**Methods**
- ML maximum likelihood
- QML quasimaximum likelihood
- MLMV maximum likelihood with missing values
- ADF asymptotic distribution free

**Techniques**
- OIM observed information matrix
- EIM expected information matrix
- OPG outer product of the gradients
- sbentler Satorra–Bentler estimator
- robust Huber/White/sandwich estimator
- clustered generalized Huber/White/sandwich estimator
- bootstrap nonparametric bootstrap
- jackknife delete-one jackknife
They are allowed in the following combinations:

<table>
<thead>
<tr>
<th>Command/Method</th>
<th>Allowed techniques</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sem</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ML</td>
<td>OIM</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>EIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OPG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sbentler</td>
<td></td>
</tr>
<tr>
<td></td>
<td>robust</td>
<td></td>
</tr>
<tr>
<td></td>
<td>clustered</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bootstrap</td>
<td></td>
</tr>
<tr>
<td></td>
<td>jackknife</td>
<td></td>
</tr>
<tr>
<td>MLMV</td>
<td>OIM</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>EIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OPG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>robust</td>
<td></td>
</tr>
<tr>
<td></td>
<td>clustered</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bootstrap</td>
<td></td>
</tr>
<tr>
<td></td>
<td>jackknife</td>
<td></td>
</tr>
<tr>
<td>ADF</td>
<td>OIM</td>
<td>default; robust-like</td>
</tr>
<tr>
<td></td>
<td>EIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bootstrap</td>
<td></td>
</tr>
<tr>
<td></td>
<td>jackknife</td>
<td></td>
</tr>
<tr>
<td><strong>gsem</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ML</td>
<td>OIM</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>OPG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>robust</td>
<td></td>
</tr>
<tr>
<td></td>
<td>clustered</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bootstrap</td>
<td></td>
</tr>
<tr>
<td></td>
<td>jackknife</td>
<td></td>
</tr>
</tbody>
</table>
Options

The corresponding options for `sem` and `gsem` to obtain each allowed method-and-technique combination are

<table>
<thead>
<tr>
<th>method()</th>
<th>vce()</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sem</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>method(ml)</td>
<td>vce(oim)</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>vce(eim)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(opg)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(sbentler)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(robust)</td>
<td>a.k.a. QML</td>
</tr>
<tr>
<td></td>
<td>vce(cluster clustvar)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(bootstrap)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(jackknife)</td>
<td></td>
</tr>
<tr>
<td>method(mlmv)</td>
<td>vce(oim)</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>vce(eim)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(opg)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(robust)</td>
<td>a.k.a. QML</td>
</tr>
<tr>
<td></td>
<td>vce(cluster clustvar)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(bootstrap)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(jackknife)</td>
<td></td>
</tr>
<tr>
<td>method(adf)</td>
<td>vce(oim)</td>
<td>default; vce(robust)-like</td>
</tr>
<tr>
<td></td>
<td>vce(eim)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(bootstrap)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(jackknife)</td>
<td></td>
</tr>
<tr>
<td><strong>gsem</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>method(ml)</td>
<td>vce(oim)</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>vce(opg)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>vce(robust)</td>
<td>a.k.a. QML</td>
</tr>
<tr>
<td></td>
<td>vce(cluster clustvar)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bootstrap</td>
<td>no option; use bootstrap: prefix</td>
</tr>
<tr>
<td></td>
<td>jackknife</td>
<td>no option; use jackknife: prefix</td>
</tr>
</tbody>
</table>

`method(emethod)` specifies the estimation method `sem` (`gsem`) is to use. If `method()` is not specified, then `method(ml)` is assumed.

`vce(vcetype)` specifies the technique to be used to obtain the VCE. When `vce()` is not specified, then `vce(oim)` is assumed.

In the case of `gsem`, `vce(bootstrap)` and `vce(jackknife)` are not allowed, although you can obtain the bootstrap or jackknife results by prefixing the `gsem` command with the `bootstrap:` or `jackknife:` prefix.

```plaintext
  . bootstrap: gsem ...
  . jackknife: gsem ...
```

See [R bootstrap] and [R jackknife]. If you are fitting a multilevel model, be sure to use `bootstrap:`'s and `jackknife:`'s `cluster()` and `idcluster()` options to obtain a correct resampling. If you have a crossed model, you cannot resample in both dimensions; there is no solution to that problem,
and therefore you cannot use the bootstrap: or jackknife: prefix. In addition, these prefixes are not allowed with models that have categorical latent variables because the categories of the latent variables (the latent classes) could change meaning from one sample to another, making it impossible to compute proper standard errors.

Remarks and examples

In [SEM] Intro 4, we gave reasons for why you might want to choose one method/technique over another. In that section, we did not draw the clear distinction between method and technique that we draw here.

The technique everyone is familiar with is OIM, although they might not know it by that name. OIM is the inverse of the negative matrix of second derivatives. Note that it is also the default for both sem and gsem.

The most common alternative techniques chosen are robust and cluster—options vce(robust) and vce(cluster clustvar). That is because they relax assumptions; see [SEM] Intro 8.

Aside: Robust and cluster are not allowed with method(adf). Nonetheless, the default OIM technique is more robust when combined with method(adf) than OIM usually is. OIM can be used even when errors are heteroskedastic. That is, the assumptions justifying the OIM calculation when combined with method(adf) are they same as those justifying vce(robust)—the errors are merely assumed to be independent.

Technique sbentler provides another version of standard errors that are robust to nonnormality when one fits a model with sem. These standard errors are a function of fourth-order moments and correspond to the Satorra–Bentler (1994) scaled chi-squared test, which is an adjustment of the model-versus-saturated goodness-of-fit statistic for nonnormal data.

Technique EIM—available with sem but not gsem—has similar properties to OIM. It is used in performing score tests. The sem command secretly calculates the EIM when necessary so that you can use postestimation score-test commands even if you estimate using a technique other than EIM. EIM is available to you because the sem command needs EIM for its own hidden purposes.

Technique OPG is not used so much anymore, although historically it was popular because it took less computer time to calculate than OIM.

For a discussion of bootstrap and jackknife variance estimation, see [R] bootstrap and [R] jackknife.

Reference


Also see

[SEM] Intro 8 — Robust and clustered standard errors
[SEM] Intro 10 — Fitting models with survey data
[SEM] gsem estimation options — Options affecting estimation
[SEM] sem option method() — Specifying method and calculation of VCE
Sometimes the data are not a simple random sample from the underlying population but instead are based on a complex survey design that can include stages of clustered sampling and stratification. Estimates produced by \texttt{sem} can be adjusted for these issues.

Adjustments for survey data are provided by \texttt{sem} and \texttt{gsem}.

Data obtained from surveys, properly treated, produce different point estimates because some observations represent a greater proportion of the underlying population than others. They also produce different standard errors because the observation-to-observation (sample-to-sample) variation is a function of the survey’s design.

To obtain survey-corrected results, you first describe the characteristics of the survey with \texttt{svyset}:

```
  . svyset county [pw=samplewgt], fpc(n_counties) strata(states) || ///
      school, fpc(n_schools)   || ///
      student, fpc(n_students)
```

In the above, we are telling Stata that our data are from a three-stage sampling design. The first stage samples without replacement counties within state; the second, schools within each sampled county; and the third, students within schools.

Once we have done that, we can tell Stata to make the survey adjustment by prefixing statistical commands with the \texttt{svy:} prefix:

```
  . svy: regress test_result teachers_per_student sex ...
```

Point estimates and standard errors will be adjusted.

You can use the \texttt{svy:} prefix with \texttt{sem} and \texttt{gsem}:

```
  . svy: sem (test_result<-...) ... (teachers_per_student->...) ...
```

See the \textit{Stata Survey Data Reference Manual} for more information on this. From a survey perspective, \texttt{sem} is not different from any other statistical command of Stata. When \texttt{gsem} is used to fit a multilevel model, stage-level sampling weights specified in the \texttt{svyset} command are applied to the corresponding hierarchical group level in the model.

Once results are estimated, you do not include the \texttt{svy:} prefix in front of the postestimation commands. You type, for instance,

```
  . estat eqtest ...
```

You do not type \texttt{svy: estat eqtest} ....

Some postestimation procedures you might ordinarily perform can be inappropriate with survey estimation results. This is because you no longer have a sample likelihood value. The postestimation command \texttt{lrttest} is an example. If you attempt to use an inappropriate postestimation command, you will be warned.
. lrtest ...

lrtest is not appropriate with survey estimation results
r(322);

Also see

[SEM] Intro 9 — Standard errors, the full story
[SEM] Intro 11 — Fitting models with summary statistics data (sem only)
[SVY] Stata Survey Data Reference Manual
Description

In textbooks and research papers, the data used are often printed in summary statistic form. These summary statistics include means, standard deviations or variances, and correlations or covariances. These summary statistics can be used in place of the underlying raw data to fit models with \texttt{sem}.

Summary statistics data (SSD) are convenient for publication because of their terseness. By not revealing individual responses, they do not violate participant confidentiality, which is sometimes important.

Support for SSD is provided by \texttt{sem} but not by \texttt{gsem}.

Remarks and examples

Remarks are presented under the following headings:

- Background
- How to use \texttt{sem} with SSD
- What you cannot do with SSD
- Entering SSD
- Entering SSD for multiple groups
- What happens when you do not set all the summary statistics
- Labeling SSD
- Making summary statistics from data for use by others

Background

The structural equation modeling estimator is a function of the first and second moments—the means and covariances—of the data. Thus it is possible to obtain estimates of the parameters of an SEM by using means and covariances. One does not need the original dataset.

In terms of \texttt{sem}, one can create a dataset containing these summary statistics and then use that dataset to obtain fitted models. The \texttt{sem} command is used just as one would use it with the original, raw data.

How to use \texttt{sem} with SSD

To use \texttt{sem} with SSD,

1. Enter the summary statistics by using the \texttt{ssd} command. How you do that is the topic of an upcoming section.
2. Save the data just as you would any dataset, namely, with the \texttt{save} command.
3. Use the \texttt{sem} command just as you would ordinarily. You use the SSD if they are not already in memory, and no special syntax or options are required by \texttt{sem}, except
a. Do not use sem’s if exp or in range qualifiers. You do not have the raw data in memory and so you cannot select subsets of the data.

b. If you have entered summary statistics for groups of observations (for example, males and, separately, females), use sem’s select() option if you want to fit the model with a subset of the groups. That is, where you would ordinarily type

```
  . sem ... if sex==1, ...
```

you instead type

```
  . sem ..., ... select(1)
```

Where you would ordinarily type

```
  . sem ... if region==1 | region==3, ...
```

you instead type

```
  . sem ..., ... select(1 3)
```

See [SEM] Example 3.

What you cannot do with SSD

With SSD in memory,

1. You cannot obtain Satorra–Bentler standard errors and the Satorra–Bentler scaled $\chi^2$ test, which you would normally do by specifying sem option vce(sbentler).

2. You cannot obtain robust standard errors, which you would normally do by specifying sem option vce(robust).

3. You cannot obtain clustered standard errors, which you would normally do by specifying sem option vce(cluster clustvar).

4. You cannot obtain survey-adjusted results, which you would normally do by specifying the svy: prefix in front of the sem command.

5. You cannot obtain bootstrap or jackknife standard errors, which you would normally do by specifying sem option vce(bootstrap) or vce(jackknife).

6. You cannot obtain VCE estimates from the observation-level outer product of the gradients, which you would normally do by specifying vce(opg).

7. You cannot use weights, which you would normally do by specifying, for instance, [fw=varname].

8. You cannot restrict the estimation sample with if exp or in range.

9. You cannot fit the model by using maximum likelihood with missing values or by using the asymptotic distribution free method, which you would normally do by specifying method(mlmv) or method(adf).

Entering SSD

Entering SSD is easy. You need to see an example of how easy it is before continuing: see [SEM] Example 2.
What follows is an outline of the procedure. Let us begin with the data you need to have. You have

1. The names of the variables. We will just call them x1, x2, and x3.
2. The number of observations, say, 74.
3. The correlations, say,

\[
\begin{pmatrix}
1 & -0.8072 & 1 \\
-0.8072 & 1 & 0.3934 \\
1 & 0.3934 & -0.5928 & 1
\end{pmatrix}
\]

or you may have the covariances,

\[
\begin{pmatrix}
33.4722 & -3.6294 & 0.6043 \\
-3.6294 & 1.0374 & -0.2120 \\
0.6043 & -0.2120 & 0.2118
\end{pmatrix}
\]

4. The variances: 33.4722, 0.6043, and 0.2118.
   Or the standard deviations: 5.6855, 0.7774, and 0.4602.
   Or neither.
   If you have the covariances in step 3, you in fact have the variances—they are just the diagonal elements of the covariance matrix—but the software will not make you enter the values twice.
5. The means: 21.2973, 3.0195, and 0.2973.
   Or not.

With that information at hand, do the following:

1. Start with no data in memory:
   \[
   . \text{clear all}
   \]
2. Initialize the SSD by stating the names of the variables:
   \[
   . \text{ssd init x1 x2 x3}
   \]

The remaining steps can be done in any order.
3. Set the number of observations:
   \[
   . \text{ssd set obs 74}
   \]
4. Set the covariances:
   \[
   . \text{ssd set cov 33.4722 \ -3.6294 \ .6043 \ 1.0374 \ -.2120 \ .2118}
   \]
   Or the correlations:
   \[
   . \text{ssd set cor 1 \ -.8072 \ 1 \ .3934 \ -.5928 \ 1}
   \]
5. If you set covariances in step 4, skip to step 6. Otherwise, if you have them, set the variances:
   \[
   . \text{ssd set var 33.4722 \ .6043 \ .2118}
   \]
   Or set the standard deviations:
   \[
   . \text{ssd set sd 5.6855 \ .7774 \ .4602}
   \]
6. Set the means if you have them:
   . ssd set means 21.2973 3.0195 .2973

7. If at any point you become confused as to what you have set and what remains to be set, type
   . ssd status

8. If you want to review what you have set, type
   . ssd list

9. If you make a mistake, you can repeat any ssd set command by adding the replace option to the end. For instance, you could reenter the means by typing
   . ssd set means 21.2973 3.0195 .2973, replace

10. Save the dataset just as you would with any dataset:
    . save mydata

You are now ready to use sem with the SSD. With the SSD in memory, you issue the sem command just as you would if you had the raw data:
    . sem ...

**Entering SSD for multiple groups**

You can enter summary statistics for groups of the data. Perhaps you have summary statistics not for the data as a whole, but for males and for females, or for the young, for the middle-aged, and for the old.

Let’s pretend you have the following data:

- **The young:**
  - observations: 74
  - correlations:
    - 1
    - 0.8072 1
    - 0.3934 −0.5928 1
  - standard deviations: 5.6855 0.7774 0.4602
  - means: 21.2973 3.0195 0.2973

- **The middle-aged:**
  - observations: 141
  - correlations:
    - 1
    - 0.5721 1
    - 0.3843 −0.4848 1
  - standard deviations: 4.9112 0.7010 0.5420
  - means: 38.1512 5.2210 0.2282

- **The old:**
  - observations: 36
  - correlations:
    - 1
    - 0.8222 1
    - 0.3712 −0.3113 1
  - standard deviations: 6.7827 0.7221 0.4305
  - means: 58.7171 2.1511 0.1623
The commands for entering these summary statistics are

```stata
.ssd init x1 x2 x3
.ssd set obs 74
.ssd set cor 1  \-.8072 1  \ .3934 -.5928 1
.ssd set sd 5.6855 .7774 .4602
.ssd set means 21.2973 3.0195 .2973
.ssd addgroup agecategory
.ssd set obs 141
.ssd set cor 1  \-.5721 1  \ .3843 -.4848 1
.ssd set sd 4.9112 .7010 .5420
.ssd set means 38.1512 5.2210 .2282
.ssd addgroup
.ssd set obs 36
.ssd set cor 1  \-.8222 1  \ .3712 -.3113 1
.ssd set sd 6.7827 .7221 .4305
.ssd set means 58.7171 2.1511 .1623
.ssd addgroup
.ssd set obs 36
.ssd set cor 1  \-.8072 1  \ .3934 -.5928 1
.ssd set sd 5.6855 .7774 .4602
.ssd set means 21.2973 3.0195 .2973
.save mygroupdata
```

The general procedure is as follows:

1. Enter the summary statistics for the first group just as outlined in the previous section.
2. Next add a group by typing
   ```stata
   . ssd addgroup newgroupvar
   ```
   In that one command, you are telling ssd that the summary statistics you entered in step 1 were for a group you are now calling `newgroupvar`, and in particular they were for `newgroupvar = 1`. You are also telling ssd that you now want to enter the summary statistics for the next group, namely, `newgroupvar = 2`.
3. Enter the summary statistics for the second group in the same way you entered them for the first group, just as outlined in the previous section.
4. If you have a third group, add it by typing
   ```stata
   . ssd addgroup
   ```
   In this case, you are telling ssd only one thing: that you now want to enter data for the next group, namely, `newgroupvar = 3`.
5. Enter the summary statistics for the third group in the same way you entered them for the second group, and just as outlined in the previous section.
6. If you want to add more groups, continue in the same way. Declare the next group of data by typing
   ```stata
   . ssd addgroup
   ```
   and then enter the data by using the ssd set command.
7. If you mistakenly add a group and wish to rescind that, type
   ```stata
   . ssd unaddgroup
   ```
8. If you wish to go back and modify the values entered for a previous group, put the group number between ssd set and what is being set—for instance, type `ssd set 2 observations ...`—and specify the replace option. For instance, to reenter the correlations for group 1, type
   ```stata
   . ssd set 1 correlations values, replace
   ```
What happens when you do not set all the summary statistics

You are required to set the number of observations and to set the covariances or the correlations. Setting the variances (standard deviations) and setting the means are optional.

1. If you set correlations only, then
   a. Means are assumed to be 0.
   b. Standard deviations are assumed to be 1.
   c. You will not be able to pool across groups if you have group data.

As a result of (a) and (b), the parameters $\text{sem}$ estimates will be standardized even when you do not specify $\text{sem}$’s standardized reporting option. Estimated means and intercepts will be 0.

Concerning (c), we need to explain. This concerns group data. If you type

\[
\text{. sem ...}
\]

then $\text{sem}$ fits a model with all the data. $\text{sem}$ does that whether you have raw data or SSD in memory. If you have SSD with groups—say, males and females or age groups 1, 2, and 3—$\text{sem}$ combines the summary statistics to obtain the summary statistics for the overall data. It is only possible to do this when covariances and means are known for each group. If you set correlations without variances or standard deviations and without means, the necessary statistics are not known and the groups cannot be combined. Thus if you type

\[
\text{. sem ...}
\]

you will get an error message. You can still estimate using $\text{sem}$; you just have to specify on which group you wish to run $\text{sem}$, and you do that with the select() option:

\[
\text{. sem ..., select(#)}
\]

2. If you set correlations and means,
   a. Standard deviations are assumed to be 1.
   b. You will not be able to pool across groups if you have group data.

This situation is nearly identical to situation 1. The only difference is that estimated means and intercepts will be nonzero.

3. If you set correlations and standard deviations or variances, or if you set covariances only,
   a. Means are assumed to be 0.
   b. You will not be able to pool across groups if you have group data.

This situation is a little better than situation 1. Estimated intercepts will be 0, but the remaining estimated coefficients will not be standardized unless you specify $\text{sem}$’s standardized reporting option.

Labeling SSD

You may use the following commands on SSD, and you use them in the same way you would with an ordinary dataset:

1. rename oldvarname newvarname
   You may rename the variables; see [D] rename.

2. label data "dataset label"
   You may label the dataset; see [D] label.
3. **label variable varname "variable label"**
   You may label variables; see [D] label.

4. **label values groupvarname valuelabelname**
   You may place a value label on the group variable; see [D] label. The group variable always takes on the values 1, 2, ....

5. **note: my note**
   **note varname: my note**
   You may place notes on the dataset or on its variables; see [D] notes.

   Do not modify the SSD except by using the ssd command. Most importantly, do not drop variables or observations.

---

**Making summary statistics from data for use by others**

If you have raw data and wish to make the summary statistics available for subsequent publication, type

```
    . ssd build varlist
```

where `varlist` lists the variables you wish to include in the dataset. The SSD will replace the raw data you had in memory. The full syntax is

```
    . ssd build varlist if exp in range
```

so you may specify `if` and `in` to restrict the observations that are included.

For instance, to build an SSD for variables `occ_prestige`, `income`, and `social_status`, type

```
    . ssd build occ_prestige income social_status
```

If you wish to build the dataset to include separate groups for males and females, type

```
    . ssd build occ_prestige income social_status, group(sex)
```

However the `sex` variable was coded in your original data, the two sexes will be now be coded 1 and 2 in the resulting SSD. Which sex is 1 and which is 2 will correspond to however `sort` would have ordered `sex` in its original coding. For instance, if variable `sex` took on values “male” and “female”, the resulting variable `sex` would take on values 1 corresponding to female and 2 corresponding to male.

Once you have built the SSD, you can describe it and list it:

```
    . ssd describe
    . ssd list
```

See [SEM] Example 25.

---

**Reference**

Also see

[SEM] Intro 10 — Fitting models with survey data
[SEM] Intro 12 — Convergence problems and how to solve them
[SEM] Example 2 — Creating a dataset from published covariances
[SEM] Example 3 — Two-factor measurement model
[SEM] Example 19 — Creating multiple-group summary statistics data
[SEM] Example 25 — Creating summary statistics data from raw data
[SEM] sem option select( ) — Using sem with summary statistics data
[SEM] ssd — Making summary statistics data (sem only)
It can be devilishly difficult for software to obtain results for SEMs. Here is what can happen:

```
. sem ...
Variables in structural equation model
(output omitted)
Fitting target model:
  initial values not feasible
  r(1400);
```

or,

```
. gsem ...
Fitting fixed-effects model:
  Iteration 0:  log likelihood = -914.65237
  Iteration 1:  log likelihood = -661.32533
  Iteration 2:  log likelihood = -657.18568
(output omitted)
Refining starting values:
  Grid node 0:  log likelihood = .
  Grid node 1:  log likelihood = .
  Grid node 2:  log likelihood = .
  Grid node 3:  log likelihood = .
Fitting full model:
  initial values not feasible
  r(1400);
```

or,

```
. sem ...
Endogenous variables
(output omitted)
Fitting target model:
  Iteration 1:  log likelihood = ...
  .
  .
  Iteration 50:  log likelihood = -337504.44  (not concave)
  Iteration 51:  log likelihood = -337503.52  (not concave)
  Iteration 52:  log likelihood = -337502.13  (not concave)
  .
  .
  Iteration 101:  log likelihood = -337400.69  (not concave)
    —Break—
  r(1);
```

In the first two cases, `sem` and `gsem` gave up. The error message is perhaps informative if not helpful. In the last case, `sem` (it could just as well have been `gsem`) iterated and iterated while producing little improvement in the log-likelihood value. We eventually tired of watching a process that was going nowhere slowly and pressed `Break`.

Now what?
Remarks and examples

Remarks are presented under the following headings:

- Is your model identified?
- Convergence solutions generically described
- Temporarily eliminate option reliability()
- Use default normalization constraints
- Temporarily eliminate feedback loops
- Temporarily simplify the model
- Try other numerical integration methods (gsem only)
- Get better starting values (sem and gsem)
- Get better starting values (gsem)

Is your model identified?

You may not know whether your model is identified. Infinite iteration logs are an indication of lack of identification or of poor starting values:

```
.semm
Endogenous variables
(output omitted)
Fitting target model:
Iteration 1: log likelihood = ...
.
.
Iteration 50: log likelihood = -337504.44 (not concave)
Iteration 51: log likelihood = -337503.52 (not concave)
Iteration 52: log likelihood = -337502.13 (not concave)
.
.
Iteration 101: log likelihood = -337400.69 (not concave)
---Break---
r(1);
```

If the problem is lack of identification, the criterion function being optimized (the log likelihood in this case) will eventually stop improving at all and yet `sem` or `gsem` will continue iterating.

If the problem is poor starting values, the criterion function will continue to increase slowly.

So if your model might not be identified, do not press `Break` too soon.

There is another way to distinguish between identification problems and poor starting values. If starting values are to blame, it is likely that a variance estimate will be heading toward 0. If the problem is lack of identification, you are more likely to see an absurd path coefficient.

To distinguish between those two alternatives, you will need to rerun the model and specify the `iterate()` option:

```
.semm ..., ... iterate(100)
(output omitted)
```

We omitted the output, but specifying `iterate(100)` allowed us to see the current parameter values at the point. We chose to specify `iterate(100)` because we knew that the likelihood function was changing slowly by that point.

If you are worried about model identification, you have a choice: Sit it out and do not press `Break` too soon, or press `Break` and rerun.
If you discover that your model is not identified, see * Identification 1: Substantive issues * in [SEM] Intro 4.

**Convergence solutions generically described**

There are three generic solutions to convergence problems that we will use:

G1. The improved-starting-values procedure:
Obtain the current parameter values from a failed attempt, modify those lousy values to make them better, use the improved values as starting values to try to fit the model again, and repeat as necessary.

G2. The alternative-starting-values procedure:
Simplify the model to produce an easier-to-fit model, fit the simplified model, use the simplified model’s solution as starting values to fit the original, more complicated model, and repeat as necessary.

G3. The alternative-software-logic procedure:
Specify strange options that make the software behave differently in hopes that a different approach will produce a solution. *sem* does not have any such strange options, but *gsem* does. In following this approach, it does not matter whether we in fact understand what we are doing because, once we find a solution, we can obtain the parameter values from the successful model and use those values as starting values to fit the model without the strange and confusing options.

Sometimes we will use all three of these procedures. We will talk about convergence and these procedures substantively below, but before we do that, we want to show you how to use a tool that you will need.

The generic solutions share a mechanical step in common, namely, obtaining parameter values from one attempt at fitting the model to use as starting values in a subsequent attempt. Here is how you do that:

```
. sem ..., ... // fit one model
. matrix b = e(b) // save parameter estimates in b
. .... // optionally modify b
. sem ..., ... from(b) // fit same model or different model
```

If the first *sem* or *gsem* command fails because you pressed *Break* or if the command issued an error message, you must reissue the command adding option *noestimate* or *iterate(#)*. Specify *noestimate* if the failure came early before the iteration log started, and otherwise specify *iterate(#)*, making # the iteration number close to but before the failure occurred:

```
. sem ..., ... noestimate
```

or

```
. sem ..., ... iterate(50)
```
Once you have obtained the parameter values in \( b \), you can list \( b \), matrix \( b = e(b) \), matrix list \( b \):

\[
\begin{array}{cccc}
  & x1: & x1: & x2: & x2: & x3: & x3: \\
  L & _\text{cons} & L & _\text{cons} & L & _\text{cons} \\
y1 & 1 & 99.518 & 1.1207418 & 99.954 & 1.1149982 & 99.052 \\
  & /: & /: & /: & /: & /: & /: \\
  \text{var}(e.x1) & \text{var}(e.x2) & \text{var}(e.x3) & \text{var}(L) \\
y1 & 123.31427 & 97.675646 & 100.72204 & 82.311401 \\
\end{array}
\]

and you can modify it:

\[
\begin{array}{cccc}
  & x1: & x1: & x2: & x2: & x3: & x3: \\
  L & _\text{cons} & L & _\text{cons} & L & _\text{cons} \\
y1 & 1 & 99.518 & 1.1207418 & 99.954 & 1.1149982 & 99.052 \\
  & /: & /: & /: & /: & /: & /: \\
  \text{var}(e.x1) & \text{var}(e.x2) & \text{var}(e.x3) & \text{var}(L) \\
y1 & 123.31427 & 97.675646 & 100.72204 & 500 \\
\end{array}
\]

And, whether you modify it or not, you can use \( b \) as the starting values for another attempt of the same model or for an attempt of a different model:

\[
. \text{sem} \ldots, \ldots \text{from}(b)
\]

**Temporarily eliminate option reliability()**

If you specified \texttt{sem}'s or \texttt{gsem}'s \texttt{reliability()} option, remove it and try fitting the model again. If the model converges, then your estimate of the reliability is too low; see \texttt{What can go wrong} in \texttt{[SEM] sem and gsem option reliability()}.

**Use default normalization constraints**

Let \texttt{sem} and \texttt{gsem} provide their default normalization constraints. By default, \texttt{sem} and \texttt{gsem} constrain all latent exogenous variables to have mean 0; constrain all latent endogenous variables to have intercept 0; and constrain the paths from latent variables to the first observed endogenous variable to have coefficient 1.

Replacing any of the above defaults can cause problems, but problems are most likely to arise if you replace the last listed default. Do not constrain path coefficients merely to obtain model identification. Let \texttt{sem} choose those constraints. It is possible that the default-coefficient-1 constraint is inappropriate for how you want to interpret your model. Relax it anyway and reimpose it later.

If default constraints solve the problem, you are done unless you want to reimpose your original, alternative constraints. That you do by typing

\[
. \text{sem} \ldots, \ldots \text{from}(b) \quad \text{// model with desired constraints}
\]

If you have multiple constraints that you want to reimpose, you may need to do them in sets.
Temporarily eliminate feedback loops

Check whether your model has any feedback loops, such as

```
sem ... (y1<-y2 x2) (y2<-y1 x3) ...
```

In this example, variable y1 affects y2 affects y1. Models with such feedback loops are said to be nonrecursive. Assume you had a solution to the above model. The results might be unstable in a substantive sense; see nonrecursive (structural) model (system) in [SEM] Glossary. The problem is that finding such truly unstable solutions is often difficult and the stability problem manifests itself as a convergence problem.

If you have convergence problems and you have feedback loops, that is not proof that the underlying values are unstable.

Regardless, temporarily remove the feedback loop,

```
sem ... (y1<-y2 x2) (y2<-x3) ...
```

and see whether the model converges. If it does, save the parameter estimates and refit the original model with the feedback, but using the saved parameter estimates as starting values.

```
matrix b = e(b)
sem ..., ... from(b)
```

If the model converges, the feedback loop is probably stable. If you are using `sem`, you can check for stability with `estat stable`. If the model does not converge, you now must find which other variables need to have starting values modified.

Temporarily simplify the model

At this point, it is difficult to know whether you should temporarily simplify your model or simply proceed with the subsequent steps and come back to simplification later should it be necessary. So proceed in whatever order seems best to you.

The idea of temporarily simplifying the model is to simplify the model, get that model to converge, and use the simplified model’s solution as starting values for the more complicated model.

The more orthogonal and independent you can make the pieces of the model, the better. Remove covariances. If you have measurement, fit it separately. Basically, remove whatever you hold most dear, because you are probably looking for subtle and correlated effects.

Once you find a simplified model that converges, do the following:

```
sem ..., ... // fit simplified model
matrix b = e(b)
sem ..., ... from(b) // fit original model
```

Try other numerical integration methods (gsem only)

For models with continuous latent variables, `gsem` provides four numerical integration methods just so you can try them. The `intmethod()` option specifies the integration method.

1. `intmethod(mvaghermite)` is the default and performs mean-and-variance adaptive Gauss–Hermite quadrature. It is fast and accurate.
2. `intmethod(mcaghermite)` performs mode-and-curvature adaptive Gauss–Hermite quadrature. It is accurate but not as fast. If you are fitting a multilevel model, there are cases where method 1 will not work but this method will work.
3. `intmethod(ghermite)` performs nonadaptive Gauss–Hermite quadrature. It is less accurate but quicker, and the calculation it makes converges more readily than either of the above methods.

4. `intmethod(laplace)` performs the Laplacian approximation instead of quadrature. It is the least accurate, sometimes the fastest, and the calculation it makes has no convergence issues whatsoever.

Try all of them. We view methods 1 and 2 as completely trustworthy. If your model will only converge with method 3 or 4, we recommend using the result as starting values for method 1 or 2. Thus you might type

```bash
  . gsem ..., ... intmethod(ghermite)
  . matrix b = e(b)
  . gsem ..., ... from(b)
```

There is another option we should mention, namely, `intpoints(#).` Methods 1, 2, and 3 default to using seven integration points. You can change that. A larger number of integration points produces more accurate results but does not improve model convergence. You might reduce the number of integration points to, say, 3. A lower number of integration points slightly improves convergence of the model, and it certainly makes model fitting much quicker. Obviously, model results are less accurate. We have been known to use fewer integration points, but mainly because of the speed issue. We can experiment more quickly. At the end, we always return to the default number of integration points.

### Get better starting values (sem and gsem)

If you observe,

```bash
  . sem ..., ...
  Variables in structural equation model
  (output omitted)
  Fitting target model:
  initial values not feasible
  r(1400);
```

and you are using `sem`, we direct you back to *Temporarily simplify the model*. The problem that we discuss here seldom reveals itself as “initial values not feasible” with `sem`. If you are using `gsem`, we direct you to the next section, *Get better starting values (gsem)*. It is not impossible that what we discuss here is the solution to your problem, but it is unlikely.
We discuss here problems that usually reveal themselves by producing an infinite iteration log:

```
  . sem ..., ...
Endogenous variables
(output omitted)
Fitting target model:
  Iteration  1:  log likelihood = ...
  .
  .
  Iteration 50:  log likelihood = -337504.44  (not concave)
  Iteration 51:  log likelihood = -337503.52  (not concave)
  Iteration 52:  log likelihood = -337502.13  (not concave)
  .
  .
  Iteration 101:  log likelihood = -337400.69  (not concave)
  —Break—
  r(1);
```

The first thing to do is look at the parameter values. To do that, type

```
  . sem ..., ... iterate(100)
```

We specified 100; you should specify an iteration value based on your log.

In most cases, you will discover that you have a variance of a latent exogenous variable going to 0, or you have a variance of an error (e.) variable going to 0.

Based on what you see, say that you suspect the problem is with the variance of the error of a latent endogenous variable \( F \) going to 0, namely, \( e.F \). You need to give that variance a larger starting value, which you can do by typing

```
  . sem ..., ... var(e.F, init(1))
```

or

```
  . sem ..., ... var(e.F, init(2))
```

or

```
  . sem ..., ... var(e.F, init(10))
```

We recommend choosing a value for the variance that is larger than you believe is necessary.

To obtain that value,

1. If the variable is observed, use `summarize` to obtain the summary statistics for the variable, square the reported standard deviation, and then increase that by, say, 20%.

2. If the variable is latent, use `summarize` to obtain a summary of the latent variable’s anchor variable and then follow the same rule: use `summarize`, square the reported standard deviation, and then increase that by 20%. (The anchor variable is the variable whose path is constrained to have coefficient 1.)

3. If the variable is latent and has paths only to other latent variables so that its anchor variable is itself latent, follow the anchor’s paths to an observed variable and follow the same rule: use `summarize`, square the reported standard deviation, and then increase that by 20%.

4. If you are using `gsem` to fit a multilevel model and the latent variable is at the observational level, follow advice 2 above. If the latent variable is at a higher level—say `school`—and its anchor is `x`, a Gaussian response with the identity link, type
. sort school  
. by school: egen avg = mean(x)  
. by school: generate touse = _n==1 if school<.  
. summarize avg if touse==1

Square the reported standard deviation and add 20%.

5. If you are using gsem and the anchor of the latent variable is not Gaussian with the identity link, see the next section.

Do not dismiss the possibility that the bad starting values concern estimated parameters other than variances of latent exogenous or error variables, although variances of those kinds of variables is the usual case. Covariances are rarely the problem because covariances can take on any value and, whether too small or too large, usually get themselves back on track as the iterative process proceeds. If you need to specify an initial value for a covariance, the syntax is

. sem ..., ..., cov(e.F*e.G, init(-25))

Substitute for $-25$ the value you consider reasonable.

The other possibility is that a path coefficient needs a better starting value, which is as unlikely as a covariance being the problem, and for the same reasons. To set the initial value of a path coefficient, add the init() option where the path is specified. Say the original sem command included $y<-x1$:

. sem ... (y<-x1 x2) ...

If you wanted to set the initial value of the path from $x1$ to 3, modify the command to read

. sem ... (y<-(x1, init(3)) x2) ...

Get better starting values (gsem)

What we say below applies regardless of how the convergence problem revealed itself. You might have seen the error message “initial values not feasible”, or some other error message, or you might have an infinite iteration log.

gsem provides two options to help you obtain better starting values: startvalues() and startgrid().

For models with continuous latent variables, startvalues(svmethod) allows you to specify one of five starting-value calculation methods: zero, constantonly, fixedonly, ivloadings, and iv. By default, gsem uses ivloadings. Evidently, that did not work for you. Try the others, starting with iv:

. gsem ..., ..., startvalues(iv)

If that does not solve the problem, proceed through the others in the following order: fixedonly, constantonly, and zero.

For models with categorical latent variables, startvalues(svmethod) allows you to specify one of seven starting-value calculation methods: factor, classid, randomid, classpr, randompr, jitter, and zero. By default, gsem uses factor.

If you have a variable, say, guessid, that provides a guess for the latent class membership of each observation, try

. gsem ..., ..., startvalues(classid guessid)

You might even use another command, such as cluster kmeans with the generate() option to create a new variable to specify in place of guessid.
The option `startvalues(randomid)` provides a function similar to `startvalues(classid guessid)` but uses random guesses for the latent class membership. You may find it helpful to try multiple random guesses and use the one with the best log likelihood after a given number of EM iterations for starting values. To do this, specify the `draws(#)` suboption. For instance, to try 10 random guesses, you can specify `startvalues(randomid, draws(10))`. You can also specify the number of EM iterations that should be taken before deciding which model is best. To do this, specify the `emopts(iterate(#))` option.

If you have a set of variables, say, `guessprlist`, that provide guesses for the probabilities for latent class membership in each observation, try

```
.gsem ..., ... startvalues(classpr guessprlist)
```

The option `startvalues(randompr)` provides a similar function but uses random guesses for the latent class probabilities. You can also specify the `draws(#)` suboption here to specify the number of random guesses that should be tried. For instance, `startvalues(randompr, draws(10))` specifies that 10 random guesses for class probabilities should be tried and that the one with the best log likelihood after the EM iterations should be used as starting values.

The option `startvalues(jitter)` applies random adjustments to starting values computed from a normal approximation to each outcome.

The option `startvalues(zero)` is really only useful if you have your own set of starting values and you want `gsem` to use zero for any model parameter without a specified value.

Whether you have continuous or categorical latent variables, if you have starting values for some parameters but not others—perhaps you fit a simplified model to get them—you can combine the options `startvalues()` and `from()`:

```
.gsem ..., ... // simplified model
.matrix b = e(b)
.gsem ..., ... from(b) startvalues(iv) // full model
```

You can combine `startvalues()` with the `init()` option, too. We described `init()` in the previous section.

The other special option `gsem` provides is `startgrid()`. This option is mostly helpful for models with continuous latent variables, but not so much for models with categorical latent variables. `startgrid()` can be used with or without `startvalues()`. `startgrid()` is a brute-force approach that tries various values for variances and covariances and chooses the ones that work best.

1. You may already be using a default form of `startgrid()` without knowing it. If you see `gsem` displaying Grid node 1, Grid node 2, ... following Grid node 0 in the iteration log, that is `gsem` doing a default search because the original starting values were not feasible. The default form tries 0.1, 1, and 10 for all variances of all latent variables, by which we mean the variances of latent exogenous variables and the variances of errors of latent endogenous variables.

2. `startgrid(numlist)` specifies values to try for variances of latent variables.

3. `startgrid(covspec)` specifies the particular variances and covariances in which grid searches are to be performed. Variances and covariances are specified in the usual way. `startgrid(e.F e.F*e.L M1[school] G*H e.y e.y1*e.y2)` specifies that 0.1, 1, and 10 be tried for each member of the list.

4. `startgrid(numlist covspec)` allows you to combine the two syntaxes, and you can specify multiple `startgrid()` options so that you can search the different ranges for different variances and covariances.
Our advice to you is this:

1. If you got an iteration log and it did not contain Grid node 1, Grid node 2, ..., then specify `startgrid(.1 1 10)`. Do that whether the iteration log was infinite or ended with some other error. In this case, we know that `gsem` did not run `startgrid()` on its own because it did not report Grid node 1, Grid node 2, etc. Your problem is poor starting values, not infeasible ones.

A synonym for `startgrid(.1 1 10)` is just `startgrid` without parentheses.

Be careful, however, if you have a large number of continuous latent variables. `startgrid` could run a long time because it runs all possible combinations. If you have 10 latent variables, that means $3^{10} = 59,049$ likelihood evaluations.

If you have a large number of continuous latent variables, rerun your difficult `gsem` command including option `iterate(#)` and look at the results. Identify the problematic variances and search across them only. Do not just look for variances going to 0. Variances getting really big can be a problem, too, and even reasonable values can be a problem. Use your knowledge and intuition about the model.

Perhaps you will try to fit your model by specifying `startgrid(.1 1 10 e.F L e.X)`. Because values 0.1, 1, and 10 are the default, you could equivalently specify `startgrid(e.F L e.X)`.

Look at covariances as well as variances. If you expect a covariance to be negative and it is positive, try negative starting values for the covariance by specifying `startgrid(-.1 -1 -10 G*H)`.

Remember that you can have multiple `startgrid()` options, and thus you could specify `startgrid(e.F L e.X) startgrid(-.1 -1 -10 G*H)`.

2. If you got “initial values not feasible”, you know that `gsem` already tried the default `startgrid`.

The default `startgrid` only tried the values 0.1, 1, and 10, and only tried them on the variances of latent variables. You may need to try different values or try the same values on covariances or variances of errors of observed endogenous variables.

We suggest you first rerun the model causing difficulty including the `noestimate` option. We also direct you back to the idea of first simplifying the model; see Temporarily simplify the model.

If, looking at the results, you have an idea of which variance or covariance is a problem, or if you have few variances and covariances, we would recommend running `startgrid()` first. On the other hand, if you have no idea as to which variance or covariance is the problem and you have a large number of them, you will be better off if you first simplify the model. If, after doing that, your simplified model does not include all the variances and covariances, you can specify a combination of `from()` and `startgrid()`.

Also see

[SEM] Intro 11 — Fitting models with summary statistics data (sem only)
Description

The SEM Builder lets you create path diagrams for SEMs, fit those models, and show results on the path diagram. Here we discuss standard linear SEMs; see [SEM] Builder, generalized for information on using the Builder to create models with generalized responses and multilevel structure.

Remarks and examples

Launch the SEM Builder by selecting the menu item Statistics > SEM (structural equation modeling) > Model building and estimation. You can also type sembuilder in the Command window.

Select the Add observed variable tool, □, and click within the diagram to add an observed variable. Type a name in the Name control of the Contextual Toolbar to name the variable. If the variable is not placed exactly where you want it, simply select the Select tool and drag the variable to your preferred location.
After adding the variable, use the Variable control in the Contextual Toolbar to select a variable from your dataset or type a variable name into the edit field.

Add latent variables to the model by using the \( \circ \) tool.

Clicking within the diagram with either the \( \square \) or \( \circ \) tool places a variable of the default size and shape. If you instead wish to create a variable of custom size and shape, click and hold the mouse button and drag until the variable is your preferred size. Select the Properties... button in the Contextual Toolbar or double-click on a variable to launch a dialog box where you can change even more properties of the variable. You can customize the size, shape, label font, and other appearance characteristics of the variable from the Appearance tab of the dialog, but you will rarely do that.

More often, you will want to change the appearance of all variables or a class of variables from the Settings menu. From the Settings menu, you can change any aspect of the appearance of variables. The Settings > Variables menu lets you change the appearance of All variables, Latent variables, Observed variables, Latent Exogenous variables, Latent Endogenous variables, Observed Exogenous variables, Observed Endogenous variables, Error variables, Latent Error variables, and Observed Error variables.

Draw paths between variables with the \( \rightarrow \) tool. Simply click in the source variable’s box or oval and drag to the target variable’s box or oval. The new path can connect variables either along the line between their center points or at the edge nearest where you click and release. This connection behavior is set in the Settings > Automation... dialog.

Whenever a path is connected to an exogenous variable, that variable becomes endogenous and an error variable is created. If you do not like the direction of the error variable relative to the new endogenous variable, first use the \( \square \) tool to select the endogenous variable and then use the buttons in the Contextual Toolbar to rotate the error variable.

Draw covariances between variables with the \( \leftarrow \) tool. If you drag to the right, the curve of the covariance will bend up. If you drag to the left, the curve will bend down. If you drag down, the curve will bend right, and if you drag up, it will bend left. That does not matter greatly; if the curve bends the opposite of what you want, simply click the Mirror button in the Contextual Toolbar.

As with variables, you can change more properties of a path or covariance (connection) by double-clicking on it or by selecting Properties from the Contextual Toolbar. You can also change the appearance of all connections from the Settings > Connections menu. From Settings > Connections, you can change the appearance of All connections, all Paths, all Covariances, or all Error Paths.

Manage variables and connections on the diagram with the Select tool, \( \mathbb{E} \). As you select an object, its Contextual Toolbar will appear. You can change the name and other properties of selected variables or the properties of selected connections. You can also drag-select or hold the Shift key and click to select multiple objects. Changing a property from the Contextual Toolbar or Properties dialog will affect all the selected objects. You can also drag selected variables as a group to other locations on the diagram.

You can place a measurement model (or measurement component of a larger model) on the diagram with the \( \mathbb{S} \) tool. Clicking on the diagram with this tool launches a dialog box where you can name the latent variable, select or type the names of the measurement variables, and specify the direction of the measurements relative to the latent variable. You can even set the spacing between the measurement variables and the distance from the latent variable to the measurements. The \( \mathbb{S} \) tool creates measurement components quickly and with even spacing. Similarly, neatly organized sets of observed variables can be placed with the \( \mathbb{O} \) tool; sets of latent variables, with the \( \mathbb{O} \) tool; and regression components, with the \( \mathbb{R} \) tool.
Place annotations and other text by using the \( \text{T} \) tool.

We have ignored the locks, \( \text{\textbullet} \), in the Contextual Toolbars. These locks apply constraints to the parameters of the SEM. You can constrain variances, \( \alpha \sigma^2 \); means, \( \mu \); intercepts, \( \alpha \alpha \); and path coefficients and covariances, \( \beta \). For example, select an exogenous variable or an error variable and type a number in \( \alpha \sigma^2 \) to constrain that variance to a fixed value. Or, select three path variables and type a name (a placeholder) in \( \beta \) to constrain all the path coefficients to be equal. You can type numbers, names, or linear expressions in the \( \text{\textbullet} \) controls. The linear expressions can involve only numbers and names that are used in other \( \text{\textbullet} \) controls.

Do not be afraid to try things. If you do not know what a tool, control, or dialog item does, try it. If you do not like the result, click on the \textbf{Undo} button, \( \text{\textleftarrow} \), in the Standard Toolbar.

Click on \( \text{\textrightarrow} \) in the Standard Toolbar to fit the model. A dialog is launched that allows you to set all the estimation options that are not defined by the path diagram. After estimation, some of the estimation results are displayed on the path diagram. Use the \textbf{Results} tab of the \textit{Settings > Variables} and \textit{Settings > Connections} dialogs to change what results are shown and how they appear (font sizes, locations, etc.). Also, as you click on connections or variables, the Details pane displays all the estimation results for the selected object.

If you wish to create another model derived from the current model, click \( \text{\textleftarrow} \) in the Standard Toolbar.

**Video example**

**SEM Builder in Stata**

Visit the Stata YouTube channel for more videos.

**Reference**

Huber, C. 2012. Using Stata’s SEM features to model the Beck Depression Inventory. \textit{The Stata Blog: Not Elsewhere Classified.} \url{http://blog.stata.com/2012/10/17/using-statass-sem-features-to-model-the-beck-depression-inventory/}. 
Builder, generalized — SEM Builder for generalized models

Description

The SEM Builder allows you to create path diagrams of generalized SEMs, fit those models, and show the results on the path diagram. Here we extend the discussion from [SEM] Builder. Read that manual entry first.

Remarks and examples

Launch the SEM Builder by selecting the menu item Statistics > SEM (structural equation modeling) > Model building and estimation. You can also type `sembuilder` in the Command window. Switch the Builder into generalized SEM mode by clicking on the Change to generalized SEM button. The generalized SEM mode allows you to add generalized responses and multilevel latent variables to your path diagram. This mode also disallows constraints on the means or variances of observed exogenous variables and on the variances of errors. In technical terms, the generalized mode draws and fits `gsem` models rather than `sem` models; see [SEM] Intro 1 for a discussion.

Aside from activating tools to add generalized responses and multilevel latent variables, everything about the Builder works the same in generalized SEM mode as it did in standard SEM mode. You can still add observed variables (□) and latent variables (○) and create connections using paths (−) and covariances (∈).

You can now add to your diagram a special kind of observed variable called a generalized response variable by using the □ tool. A generalized response is a dependent variable that may be continuous, count, binary, ordinal, or multinomial, and whose statistical properties are described by its family and link; see [SEM] Intro 2 for a discussion. Simply click on the □ tool and then click in the diagram to add a generalized response variable. After adding the generalized response, use the Contextual Toolbar to do the following: 1) select its Family/Link and 2) select its variable from the Variable control. You may also type a variable name into the Variable control.

You can also add multilevel latent variables—random effects assumed to have a Gaussian distribution whose realizations are at the level of a grouping variable or variables. See [SEM] Intro 2 for a discussion. Click on the ○ tool and then click on the diagram to add a multilevel latent variable. After adding the variable, click on the " button of the Name/Level control in the Contextual Toolbar. In the resulting dialog box, choose the nesting depth of your latent variable, and then choose the variables that define the nesting. For example, for a three-level model of doctors nesting patients nesting observations, you would select 3 in the Nesting depth control, select doctor in the first variable control, and select patient in the second variable control, leaving > Observations, which is fixed. You do not need to change the default Base name, but you may type a different name of your choice. Do read the note on the dialog carefully because it provides guidance on successfully creating multilevel models. Also see [SEM] Example 39g for an example that demonstrates creating a nested model in the Builder.
As with standard observed and latent variables, you can also click and drag as you add generalized responses and multilevel latent variables to control the size of the resulting variables. You can also select the Properties... button from the Contextual Toolbar or double-click on a variable to change even more properties of the variable. As with standard variables, you can control their appearance from the Appearance tab of the resulting dialog box or, more likely, control the appearance of all variables of a given type from the Settings > Variables menu.

As with standard models, you create the structure of your model by connecting the variables with the tool. You may also specify covariances among variables with the tool.

There is one other construct available in generalized mode that is not available in standard mode—paths to other paths. In addition to creating paths from a latent variable to an observed variable or to another latent variable at the same nesting level, you can connect a standard or multilevel latent variable to a path from one observed variable to another observed variable. This makes that path a random path, or a random slope. As with other path connections, think of the path as adding the source variable to the target. So we are adding a random variable to a path (or slope), making it a random path. See [SEM] Example 38g for an example of creating a random path in the Builder.

In generalized mode, the Add Measurement Component tool, allows the measurements to be created as generalized responses. Simply click on the Make measurements generalized box on the dialog box and select a Family/Link. It also allows the latent variable to be either standard observation-level or multilevel with the radio control at the top of the dialog box.

The Add observed variables set tool, allows you to create multiple generalized responses. Click on the Make variables generalized responses box on the dialog box and select a Family/Link.

Creating multinomial logit responses can be tricky. You must create a generalized response for each level of your outcome and use factor-variable notation to designate the levels of the outcome. The tool is the safest way to create a multinomial logit response. With your dataset in memory, select the tool and then click on the diagram. Click on the Make variables generalized responses box on the dialog box, and then select Multinomial, Logit in the Family/Link control. Select your multinomial response variable from the Variable control, and the levels for your multinomial response will be automatically populated. The level designated with a \( b \) will be the base level. You can type the \( b \) on another level to make it the base. When you click OK, multinomial generalized responses will be created for each level of your variable. See [SEM] Example 37g for an example of creating multinomial logit responses in the Builder.

The Add Regression Component tool, also allows the dependent variable to be specified as a generalized response.

Place shaded boxes on the diagram with the tool. This tool can be used in both standard and generalized mode. It is most useful in the generalized mode, where we can shade different nesting levels of multilevel models.

As in standard mode, constraints are specified with lock controls *, and you fit the model by using the button on the Standard Toolbar.

Video example

SEM Builder in Stata

Visit the Stata YouTube channel for more videos.
Reference

**estat eform — Display exponentiated coefficients**

### Description

`estat eform` is for use after `gsem` but not `sem`.

`gsem` reports coefficients. You can obtain exponentiated coefficients and their standard errors by using `estat eform` after estimation to redisplay results.

### Menu

Statistics > SEM (structural equation modeling) > Other > Display exponentiated coefficients

### Syntax

```
estat eform [eqnamelist] [ , level(#) display_options ]
```

where `eqnamelist` is a list of equation names. In `gsem`, equation names correspond to the names of the response variables. If no `eqnamelist` is specified, exponentiated results for the first equation are shown.

### Options

`level(#);` see [R] Estimation options; default is `level(95)`

`display_options` control the display of factor variables and more. Allowed `display_options` are `noci`, `nopvalues`, `nomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(style)`, `cformat(%fmt)`, `pformat(%fmt)`, `sformat(%fmt)`, and `nolstretch`. See [R] Estimation options.

### Remarks and examples

In some generalized linear response functions, exponentiated coefficients have a special meaning. Those special meanings are as follows:

<table>
<thead>
<tr>
<th>Common name</th>
<th>Family</th>
<th>Link</th>
<th>Meaning of exp(coef)</th>
</tr>
</thead>
<tbody>
<tr>
<td>logit</td>
<td>Bernoulli</td>
<td>logit</td>
<td>odds ratio</td>
</tr>
<tr>
<td>ologit</td>
<td>ordinal</td>
<td>logit</td>
<td>odds ratio</td>
</tr>
<tr>
<td>mlogit</td>
<td>multinomial</td>
<td>logit</td>
<td>relative-risk ratio</td>
</tr>
<tr>
<td>Poisson</td>
<td>Poisson</td>
<td>log</td>
<td>incidence-rate ratio</td>
</tr>
<tr>
<td>nbreg</td>
<td>nbreg</td>
<td>log</td>
<td>incidence-rate ratio</td>
</tr>
</tbody>
</table>
### Survival distribution

<table>
<thead>
<tr>
<th>Survival Distribution</th>
<th>Meaning of exp(coef)</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential</td>
<td>hazard ratio</td>
</tr>
<tr>
<td>Weibull</td>
<td>hazard ratio</td>
</tr>
<tr>
<td>gamma</td>
<td>time ratio</td>
</tr>
<tr>
<td>loglogistic</td>
<td>time ratio</td>
</tr>
<tr>
<td>lognormal</td>
<td>time ratio</td>
</tr>
</tbody>
</table>

See [SEM] Example 33g and [SEM] Example 34g.

### Also see

[SEM] gsem — Generalized structural equation model estimation command

[SEM] gsem postestimation — Postestimation tools for gsem

[SEM] Intro 7 — Postestimation tests and predictions

[SEM] Example 33g — Logistic regression

[SEM] Example 34g — Combined models (generalized responses)

[SEM] Example 47g — Exponential survival model

[SEM] Example 48g — Loglogistic survival model with censored and truncated data
**Description**

`estat eqgof` is for use after `sem` but not `gsem`.

`estat eqgof` displays equation-by-equation goodness-of-fit statistics. Displayed are $R^2$ and the Bentler–Raykov squared multiple-correlation coefficient (Bentler and Raykov 2000).

These two concepts of fit are equivalent for recursive SEMs and univariate linear regression. For nonrecursive SEMs, these measures are distinct.

Equation-level variance decomposition is also reported, along with the overall model coefficient of determination.

**Menu**

Statistics $>$ SEM (structural equation modeling) $>$ Goodness of fit $>$ Equation-level goodness of fit

**Syntax**

```
estat eqgof [, format(%fmt)]
```

**Option**

`format(%fmt)` specifies the display format. The default is `format(%9.0f)`.

**Remarks and examples**

See [SEM] Example 3.

In rare circumstances, these equation-level goodness-of-fit measures in nonrecursive structural equations have unexpected values. It is possible to obtain negative $R^2$ and multiple-correlation values.

It is recommended to use the Bentler–Raykov squared multiple correlations as a measure of explained variance for nonrecursive systems that involve endogenous variables with reciprocal causations.

**Stored results**

`estat eqgof` stores the following in `r()`:

Scalars

- `r(N_groups)`: number of groups
- `r(CD[#])`: overall coefficient of determination (for group #)

Matrices

- `r(nobs)`: sample size for each group
- `r(eqfit[#])`: fit statistics (for group #)
Reference


Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM] `sem postestimation` — Postestimation tools for sem
- [SEM] `estat gof` — Goodness-of-fit statistics
- [SEM] `estat ggof` — Group-level goodness-of-fit statistics
- [SEM] Example 3 — Two-factor measurement model
- [SEM] Methods and formulas for sem — Methods and formulas for sem
**Title**

| estat eqtest — Equation-level tests that all coefficients are zero |

**Description**

*estat eqtest* is for use after *sem* but not *gsem*.

*estat eqtest* displays Wald tests that all coefficients excluding the intercept are 0 for each equation in the model.

**Menu**

Statistics > SEM (structural equation modeling) > Testing and CIs > Equation-level Wald tests

**Syntax**

```
estat eqtest [, total nosvyadjust]
```

**Options**

`total` is for use when estimation was with *sem, group()*. It specifies that the tests be aggregated across the groups.

`nosvyadjust` is for use with *svy* estimation commands. It specifies that the Wald tests be carried out without the default adjustment for the design degrees of freedom. That is to say the tests are carried out as $W/k \sim F(k, d)$ rather than as $(d - k + 1)W/(kd) \sim F(k, d - k + 1)$, where $k$ is the dimension of the tests and $d$ is the total number of sampled PSUs minus the total number of strata.

**Remarks and examples**

See [SEM] Example 13.

**Stored results**

*estat eqtest* stores the following in *r()*:

Scalars

- `r(N_groups)`: number of groups

Matrices

- `r(nobs)`: sample size for each group
- `r(test[#])`: test statistics (for group #)
- `r(test_total)`: aggregated test statistics (total only)
Also see

[SEM] `sem` — Structural equation model estimation command
[SEM] `sem postestimation` — Postestimation tools for sem
[SEM] `lrtest` — Likelihood-ratio test of linear hypothesis
[SEM] `test` — Wald test of linear hypotheses
[SEM] Example 13 — Equation-level Wald test
[SEM] Methods and formulas for sem — Methods and formulas for sem
Title

\textbf{estat framework} — Display estimation results in modeling framework

\begin{tabular}{l|l}
\hline
\multicolumn{2}{l}{Description} \\
\multicolumn{2}{l}{Remarks and examples} \\
\multicolumn{2}{l}{Menu} \\
\multicolumn{2}{l}{Syntax} \\
\multicolumn{2}{l}{Options} \\
\hline
\end{tabular}

\section*{Description}

\texttt{estat framework} is a postestimation command for use after \texttt{sem} but not \texttt{gsem}.

\texttt{estat framework} displays the estimation results as a series of matrices derived from the Bentler–Weeks form; see Bentler and Weeks (1980).

\section*{Menu}

Statistics \textgreater{} SEM (structural equation modeling) \textgreater{} Other \textgreater{} Report model framework

\section*{Syntax}

\texttt{estat framework} [ , \textit{options} ]

\begin{tabular}{l|l}
\hline
\textit{options} & Description \\
\hline
\texttt{standardized} & report standardized results \\
\texttt{compact} & display matrices in compact form \\
\texttt{fitted} & include fitted means, variances, and covariances \\
\texttt{format(\%fmt)} & display format to use \\
\hline
\end{tabular}

\section*{Options}

\texttt{standardized} reports results in standardized form.

\texttt{compact} displays matrices in compact form. Zero matrices are displayed as a description. Diagonal matrices are shown as a row vector.

\texttt{fitted} displays the fitted mean and covariance values.

\texttt{format(\%fmt)} specifies the display format to be used. The default is \texttt{format(\%9.0g)}.

\section*{Remarks and examples}

See \texttt{[SEM] Example 11}.
Technical note

If `sem`’s `nm1` option was specified when the model was fit, all covariance matrices are calculated using $N - 1$ in the denominator instead of $N$.

## Stored results

`estat framework` stores the following in `r()`:

### Scalars
- $r(N_{\text{groups}})$: number of groups
- $r(\text{standardized})$: indicator of standardized results (+)

### Matrices
- $r(\text{nobs})$: sample size for each group
- $r(\text{Beta}[\_\#])$: coefficients of endogenous variables on endogenous variables (for group #)
- $r(\text{Gamma}[\_\#])$: coefficients of endogenous variables on exogenous variables (for group #)
- $r(\text{alpha}[\_\#])$: intercepts (for group #) (*)
- $r(\text{Psi}[\_\#])$: covariances of errors (for group #)
- $r(\text{Phi}[\_\#])$: covariances of exogenous variables (for group #)
- $r(\text{kappa}[\_\#])$: means of exogenous variables (for group #) (*)
- $r(\text{Sigma}[\_\#])$: fitted covariances (for group #)
- $r(\text{mu}[\_\#])$: fitted means (for group #) (*)

(+) If $r(\text{standardized})=1$, the returned matrices contain standardized values.

(*) If there are no estimated means or intercepts in the `sem` model, these matrices are not returned.

## Reference


## Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM] `sem postestimation` — Postestimation tools for `sem`
- [SEM] `Intro 7` — Postestimation tests and predictions (*Replaying the model (sem and gsem)*)
- [SEM] `Intro 7` — Postestimation tests and predictions (*Accessing stored results*)
- [SEM] `Example 11` — estat framework
- [SEM] `Methods and formulas for sem` — Methods and formulas for `sem`
**Description**

`estat ggof` is for use after estimation with `sem, group()`. `estat ggof` displays, by group, the standardized root mean squared residual (SRMR), the coefficient of determination (CD), and the model versus saturated $\chi^2$ along with its associated degrees of freedom and $p$-value.

**Menu**

Statistics > SEM (structural equation modeling) > Group statistics > Group-level goodness of fit

**Syntax**

```
estat ggof [ , format(%fmt) ]```

**Option**

`format(%fmt)` specifies the display format. The default is `format(%9.3f)`.

**Remarks and examples**

See [SEM] Example 21.

`estat ggof` provides group-level goodness-of-fit statistics after estimation by `sem, group()`; see [SEM] `sem group options`.

The SRMR, CD, and $\chi^2$ statistics are not computed for models fit by `gsem`; therefore, `estat ggof` is not for use after estimation with `gsem, group()`.

**Stored results**

`estat ggof` stores the following in `r()`:

Scalars

- `r(N_groups)` number of groups

Matrices

- `r(gfit)` fit statistics
- `r(gfit_sb)` Satorra–Bentler scaled fit statistics
Also see

SEM | sem — Structural equation model estimation command
SEM | sem postestimation — Postestimation tools for sem
SEM | estat eqgof — Equation-level goodness-of-fit statistics
SEM | estat gof — Goodness-of-fit statistics
SEM | Example 21 — Group-level goodness of fit
SEM | Methods and formulas for sem — Methods and formulas for sem
SEM | sem group options — Fitting models on different groups
**estat ginvariant — Tests for invariance of parameters across groups**

**Description**

`estat ginvariant` is for use after estimation with `sem, group()`; see [SEM] `sem group options`.

`estat ginvariant` performs score tests (Lagrange multiplier tests) and Wald tests of whether parameters constrained to be equal across groups should be relaxed and whether parameters allowed to vary across groups could be constrained.


**Menu**

Statistics > SEM (structural equation modeling) > Group statistics > Test invariance of parameters across groups

**Syntax**

```
estat ginvariant [, options]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>showpclass(pclassname)</code></td>
<td>restrict output to parameters in the specified parameter class</td>
</tr>
<tr>
<td><code>class</code></td>
<td>include joint tests for parameter classes</td>
</tr>
<tr>
<td><code>legend</code></td>
<td>include legend describing parameter classes</td>
</tr>
</tbody>
</table>

**pclassname**

<table>
<thead>
<tr>
<th>pclassname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>scoef</code></td>
<td>structural coefficients</td>
</tr>
<tr>
<td><code>scons</code></td>
<td>structural intercepts</td>
</tr>
<tr>
<td><code>mcoef</code></td>
<td>measurement coefficients</td>
</tr>
<tr>
<td><code>mcons</code></td>
<td>measurement intercepts</td>
</tr>
<tr>
<td><code>serrvar</code></td>
<td>covariances of structural errors</td>
</tr>
<tr>
<td><code>merrvar</code></td>
<td>covariances of measurement errors</td>
</tr>
<tr>
<td><code>smerrcov</code></td>
<td>covariances between structural and measurement errors</td>
</tr>
<tr>
<td><code>meanex</code></td>
<td>means of exogenous variables</td>
</tr>
<tr>
<td><code>covex</code></td>
<td>covariances of exogenous variables</td>
</tr>
<tr>
<td><code>all</code></td>
<td>all the above</td>
</tr>
<tr>
<td><code>none</code></td>
<td>none of the above</td>
</tr>
</tbody>
</table>
Options

showpclass(classname) displays tests for the classes specified. showpclass(all) is the default. class displays a table with joint tests for group invariance for each of the nine parameter classes. legend displays a legend describing the parameter classes. This option may only be used with the class option.

Remarks and examples

See [SEM] Example 22.

Score tests are not available after gsem; therefore, estat ginvariant is not for use after estimation with gsem, group().

Stored results

estat ginvariant stores the following in r():

Scalars
    r(N_groups) number of groups

Matrices
    r(nobs) sample size for each group
    r(test) Wald and score tests
    r(test_pclass) parameter classes corresponding to r(test)
    r(test_class) joint Wald and score tests for each class

References


Also see

[SEM] sem — Structural equation model estimation command
[SEM] sem postestimation — Postestimation tools for sem
[SEM] estat mindices — Modification indices
[SEM] estat scoretests — Score tests
[SEM] Example 22 — Testing parameter equality across groups
[SEM] Methods and formulas for sem — Methods and formulas for sem
estat gof — Goodness-of-fit statistics

Description

estat gof is for use after \texttt{sem} but not \texttt{gsem}.
estat gof displays a variety of overall goodness-of-fit statistics.

Menu

Statistics $>$ SEM (structural equation modeling) $>$ Goodness of fit $>$ Overall goodness of fit

Syntax

\texttt{estat gof} [ \textit{, options} ]

\begin{tabular}{ll}
\textit{options} & Description \\
\hline
\texttt{stats(statlist)} & statistics to be displayed \\
\texttt{nose} & \texttt{describe} suppress descriptions of statistics \\
\hline
\texttt{statlist} & Description \\
\texttt{chi2} & $\chi^2$ tests; the default \\
\texttt{rmsea} & root mean squared error of approximation \\
\texttt{ic} & information indices \\
\texttt{indices} & indices for comparison against baseline \\
\texttt{residuals} & measures based on residuals \\
\texttt{all} & all the above \\
\end{tabular}

Note: The statistics reported by \texttt{chi2}, \texttt{rmsea}, and \texttt{indices} are dependent on the assumption of joint normality of the observed variables. If \texttt{vce(sbentler)} is specified with \texttt{sem}, modified versions of these statistics that are computed using the Satorra–Bentler scaled $\chi^2$ statistics will also be reported.
Options

\texttt{stats(statlist)} specifies the statistics to be displayed. The default is \texttt{stats(chi2)}.

\texttt{stats(chi2)} reports the model versus saturated test and the baseline versus saturated test. The saturated model is the model that fits the covariances perfectly.

The model versus saturated test is a repeat of the test reported at the bottom of the \texttt{sem} output. In the baseline versus saturated test, the baseline model includes the means and variances of all observed variables plus the covariances of all observed exogenous variables. For a covariance model (a model with no endogenous variables), the baseline includes only the means and variances of observed variables. Be aware that different authors define the baseline model differently.

\texttt{stats(rmsea)} reports the root mean squared error of approximation (RMSEA) and its 90\% confidence interval, and pclose, the \textit{p}-value for a test of close fit, namely, RMSEA < 0.05. Most interpreters of this test label the fit close if the lower bound of the 90\% CI is below 0.05 and label the fit poor if the upper bound is above 0.10. See Browne and Cudeck (1993).

\texttt{stats(ic)} reports the Akaike information criterion (AIC) and Bayesian (or Schwarz) information criterion (BIC). These statistics are available only after estimation with \texttt{sem method(ml)} or \texttt{method(mlmv)}. These statistics are used not to judge fit in absolute terms but instead to compare the fit of different models. Smaller values indicate a better fit. Be aware that there are many variations (minor adjustments) to statistics labeled AIC and BIC. Reported here are statistics that match \texttt{estat ic}; see \texttt{[R] estat ic}.

To compare models that use statistics based on likelihoods, such as AIC and BIC, models should include the same variables; see \texttt{[SEM] lrtest}. See Akaike (1987), Schwarz (1978), and Raftery (1993).

\texttt{stats(indices)} reports CFI and TLI, two indices such that a value close to 1 indicates a good fit. CFI stands for comparative fit index. TLI stands for Tucker–Lewis index and is also known as the nonnormed fit index. See Bentler (1990).

\texttt{stats(residuals)} reports the standardized root mean squared residual (SRMR) and the coefficient of determination (CD).

A perfect fit corresponds to an SRMR of 0. A good fit is a small value, considered by some to be limited to 0.08.

Concerning CD, a perfect fit corresponds to a CD of 1. CD is like \textit{R}^2 for the whole model.

\texttt{stats(all)} reports all the statistics. You can also specify just the statistics you wish reported, such as

\begin{verbatim}
. estat gof, stats(indices residuals)
\end{verbatim}

\texttt{nодescribe} suppresses the descriptions of the goodness-of-fit measures.

Remarks and examples

See \texttt{[SEM] Example 4}.
Stored results

`estat gof` stores the following in `r()`:

### Scalars
- `r(chi2_ms)`: test of target model against saturated model
- `r(df_ms)`: degrees of freedom for `r(chi2_ms)`
- `r(p_ms)`: p-value for `r(chi2_ms)`
- `r(chi2sb_ms)`: Satorra–Bentler scaled test of target model against saturated model
- `r(psb_ms)`: p-value for `r(chi2sb_ms)`
- `r(chi2_bs)`: test of baseline model against saturated model
- `r(df_bs)`: degrees of freedom for `r(chi2_bs)`
- `r(p_bs)`: p-value for `r(chi2_bs)`
- `r(chi2sb_bs)`: Satorra–Bentler scaled test of baseline model against saturated model
- `r(psb_bs)`: p-value for `r(chi2sb_bs)`
- `r(rmsea)`: root mean squared error of approximation
- `r(lb90_rmsea)`: lower bound of 90% CI for RMSEA
- `r(ub90_rmsea)`: upper bound of 90% CI for RMSEA
- `r(pclose)`: p-value for test of close fit: RMSEA < 0.05
- `r(rmsea_sb)`: RMSEA using Satorra–Bentler $\chi^2$
- `r(aic)`: Akaike information criterion
- `r(bic)`: Bayesian information criterion
- `r(cfi)`: comparative fit index
- `r(cfi_sb)`: CFI using Satorra–Bentler $\chi^2$
- `r(tli)`: Tucker–Lewis fit index
- `r(tli_sb)`: TLI using Satorra–Bentler $\chi^2$
- `r(cd)`: coefficient of determination
- `r(srmr)`: standardized root mean squared residual
- `r(N_groups)`: number of groups

### Matrices
- `r(nobs)`: sample size for each group

### References


### Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM] `sem postestimation` — Postestimation tools for sem
- [SEM] `estat eqgof` — Equation-level goodness-of-fit statistics
- [SEM] `estat ggof` — Group-level goodness-of-fit statistics
- [SEM] `estat residuals` — Display mean and covariance residuals
- [SEM] `Example 4` — Goodness-of-fit statistics
- [SEM] `Methods and formulas for sem` — Methods and formulas for sem
- [R] `estat ic` — Display information criteria
estat lcgof — Latent class goodness-of-fit statistics

Description

estat lcgof is for use after gsem but not sem.
estat lcgof displays a variety of overall goodness-of-fit statistics for latent class models.

Menu

Statistics > LCA (latent class analysis) > Goodness of fit

Syntax

estat lcgof [, nodescribe]

Option

nodescribe suppresses the descriptions of the goodness-of-fit measures.

Remarks and examples

estat lcgof reports AIC and BIC for the fitted model.

For standard latent class models, estat lcgof also reports a likelihood-ratio test of the fitted model versus the saturated model. The likelihood-ratio statistic is also known as the $G^2$ statistic.

See [SEM] Example 51g.

Stored results

estat lcgof stores the following in r():

Scalars

- r(chi2_ms) test of target model against saturated model
- r(df_ms) degrees of freedom for r(chi2_ms)
- r(p_ms) $p$-value for r(chi2_ms)
- r(aic) Akaike information criterion
- r(bic) Bayesian information criterion
References


Also see

[SEM] gsem — Generalized structural equation model estimation command

[SEM] gsem postestimation — Postestimation tools for gsem

[SEM] Example 51g — Latent class goodness-of-fit statistics

[R] estat ic — Display information criteria
**estat lcmean — Latent class marginal means**

**Description**

`estat lcmean` is for use after `gsem` but not `sem`.

`estat lcmean` reports a table of the marginal predicted means of each outcome within each latent class.

**Menu**

Statistics > LCA (latent class analysis) > Class marginal means

**Syntax**

`estat lcmean [, options]`

**Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nose</td>
<td>do not estimate SEs</td>
</tr>
<tr>
<td>post</td>
<td>post margins and their VCE as estimation results</td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control column formats, row spacing, and line width</td>
</tr>
</tbody>
</table>

**Remarks and examples**

See [SEM] Example 50g, [SEM] Example 53g, and [SEM] Example 54g.
Stored results

`estat lcmean` stores the following in `r()`:

Scalars
- `r(N)` number of observations

Macros
- `r(title)` title in output

Matrices
- `r(b)` estimates
- `r(V)` variance–covariance matrix of the estimates
- `r(table)` matrix containing the margins with their standard errors, test statistics, $p$-values, and confidence intervals

`estat lcmean` with the `post` option also stores the following in `e()`:

Scalars
- `e(N)` number of observations

Macros
- `e(title)` title in output
- `e(properties) b V`

Matrices
- `e(b)` estimates
- `e(V)` variance–covariance matrix of the estimates

Also see

- [SEM] `gsem` — Generalized structural equation model estimation command
- [SEM] `gsem postestimation` — Postestimation tools for gsem
- [SEM] Example 50g — Latent class model
- [SEM] Example 53g — Finite mixture Poisson regression
- [SEM] Example 54g — Finite mixture Poisson regression, multiple responses
estat lcprob — Latent class marginal probabilities

Description

estat lcprob is for use after gsem but not sem.
estat lcprob reports a table of the marginal predicted latent class probabilities.

Menu

Statistics > LCA (latent class analysis) > Class marginal probabilities

Syntax

estat lcprob [ , options ]

options          Description

classpr           latent class probability; the default
classtoposteriorpr posterior latent class probability
nose              do not estimate SEs
post              post margins and their VCE as estimation results
display_options  control column formats, row spacing, and line width

Options

classpr, the default, calculates marginal predicted probabilities for each latent class.
classtoposteriorpr calculates marginal predicted posterior probabilities for each latent class. The posterior probabilities are a function of the latent class predictors and the fitted outcome densities.
nose suppresses calculation of the VCE and standard errors.
post causes estat lcprob to behave like a Stata estimation (e-class) command. estat lcprob posts the vector of estimated margins along with the estimated variance–covariance matrix to e(), so you can treat the estimated margins just as you would results from any other estimation command.
display_options: vsquish, fvwrap(#), fvwrapon(style), cformat(‘fmt’), pformat(‘fmt’), sformat(‘fmt’), and nolstretch.
Remarks and examples

See [SEM] Example 50g, [SEM] Example 53g, and [SEM] Example 54g.

Stored results

`estat lcprob` stores the following in `r()`:

Scalars
- `r(N)`: number of observations

Macros
- `r(title)`: title in output
- `r(classposteriorpr)`: classposteriorpr

Matrices
- `r(b)`: estimates
- `r(V)`: variance–covariance matrix of the estimates
- `r(table)`: matrix containing the margins with their standard errors, test statistics, *p*-values, and confidence intervals

`estat lcprob` with the `post` option also stores the following in `e()`:

Scalars
- `e(N)`: number of observations

Macros
- `e(title)`: title in output
- `e(classposteriorpr)`: classposteriorpr
- `e(properties)`: `b V`

Matrices
- `e(b)`: estimates
- `e(V)`: variance–covariance matrix of the estimates

Also see

[SEM] `gsem` — Generalized structural equation model estimation command
[SEM] `gsem postestimation` — Postestimation tools for `gsem`
[SEM] Example 50g — Latent class model
[SEM] Example 53g — Finite mixture Poisson regression
[SEM] Example 54g — Finite mixture Poisson regression, multiple responses
estat mindices — Modification indices

Description

estat mindices is for use after sem but not gsem.

estat mindices reports modification indices for path coefficients and covariances that were constrained or omitted in the fitted model. Modification indices are score tests (Lagrange multiplier tests) for the statistical significance of the constrained parameters. See Sörbom (1989) and Wooldridge (2010, 421–428).

Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Modification indices

Syntax

estat mindices [, options ]

options Description

showpclass(pclassname) restrict output to parameters in the specified parameter classes

minchi2(#) display only tests with modification index (MI) ≥ #

pclassname Description

scoef structural coefficients
scons structural intercepts
mcoef measurement coefficients
mcons measurement intercepts
serrvar covariances of structural errors
merrvar covariances of measurement errors
smerrcov covariances between structural and measurement errors
meanex means of exogenous variables
covex covariances of exogenous variables
all all the above
none none of the above
Options

\texttt{showpclass(pclassname)} specifies that results be limited to parameters that belong to the specified parameter classes. The default is \texttt{showpclass(all)}.

\texttt{minchi2(#)} suppresses listing paths with modification indices (MIs) less than \#. By default, \texttt{estat mindices} lists values significant at the 0.05 level, corresponding to $\chi^2(1)$ value \texttt{minchi2(3.8414588)}. Specify \texttt{minchi2(0)} if you wish to see all tests.

Remarks and examples

See \citeindex{SEM} Example 5.

Stored results

\texttt{estat mindices} stores the following in \texttt{r()}:  

\begin{itemize}
  \item Scalars
    \begin{itemize}
      \item \texttt{r(N_groups)}  \hspace{1cm} \text{number of groups}
    \end{itemize}
  \item Matrices
    \begin{itemize}
      \item \texttt{r(nobs)}  \hspace{1cm} \text{sample size for each group}
      \item \texttt{r(mindices_pclass)}  \hspace{1cm} \text{parameter class of modification indices}
      \item \texttt{r(mindices)}  \hspace{1cm} \text{matrix containing the displayed table values}
    \end{itemize}
\end{itemize}

References


Also see

\citeindex{SEM} \texttt{sem} — Structural equation model estimation command

\citeindex{SEM} \texttt{sem postestimation} — Postestimation tools for \texttt{sem}

\citeindex{SEM} \texttt{estat ginvariant} — Tests for invariance of parameters across groups

\citeindex{SEM} \texttt{estat scoretests} — Score tests

\citeindex{SEM} Example 5 — Modification indices

\citeindex{SEM} Methods and formulas for \texttt{sem} — Methods and formulas for \texttt{sem}
**Title**

| estat residuals — Display mean and covariance residuals |

**Description**

`estat residuals` is for use after `sem` but not `gsem`.

`estat residuals` displays the mean and covariance residuals. Normalized and standardized residuals are available.

Both mean and covariance residuals are reported unless `sem`’s option `nomeans` was specified or implied at the time the model was fit, in which case mean residuals are not reported.

`estat residuals` usually does not work following `sem` models fit with `method(mlmv)`. It also does not work if there are any missing values, which after all is the whole point of using `method(mlmv)`.

**Menu**

Statistics > SEM (structural equation modeling) > Goodness of fit > Matrices of residuals

**Syntax**

```plaintext
estat residuals [, options]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>normalized</td>
<td>report normalized residuals</td>
</tr>
<tr>
<td>standardized</td>
<td>report standardized residuals</td>
</tr>
<tr>
<td>sample</td>
<td>use sample covariances in residual variance calculations</td>
</tr>
<tr>
<td>nm1</td>
<td>use adjustment $N - 1$ in residual variance calculations</td>
</tr>
<tr>
<td>zerotolerance(tol)</td>
<td>apply tolerance to treat residuals as 0</td>
</tr>
<tr>
<td>format(%,fmt)</td>
<td>display format</td>
</tr>
</tbody>
</table>

**Options**

`normalized` and `standardized` are alternatives. If neither is specified, raw residuals are reported.

Normalized residuals and standardized residuals attempt to adjust the residuals in the same way, but they go about it differently. The normalized residuals are always valid, but they do not follow a standard normal distribution. The standardized residuals do follow a standard normal distribution but only if they can be calculated; otherwise, they will equal missing values. When both can be calculated (equivalent to both being appropriate), the normalized residuals will be a little smaller than the standardized residuals. See Jöreskog and Sörbom (1986).
sample specifies that the sample variance and covariances be used in variance formulas to compute normalized and standardized residuals. The default uses fitted variance and covariance values as described by Bollen (1989).

nm1 specifies that the variances be computed using $N - 1$ in the denominator rather than using sample size $N$.

zerotolerance(tol) treats residuals within tol of 0 as if they were 0. tol must be a numeric value less than 1. The default is zerotolerance(0), meaning that no tolerance is applied. When standardized residuals cannot be calculated, it is because a variance calculated by the Hausman (1978) theorem turns negative. Applying a tolerance to the residuals turns some residuals into 0 and then division by the negative variance becomes irrelevant, and that may be enough to solve the calculation problem.

format(%fmt) specifies the display format. The default is format(%.9.3f).

Remarks and examples

See [SEM] Example 10.

Stored results

estat residuals stores the following in r():

Scalars
- $r(N\_groups)$: number of groups

Macros
- $r(sample)$: empty or sample, if sample was specified
- $r(nm1)$: empty or nm1, if nm1 was specified

Matrices
- $r(nobs)$: sample size for each group
- $r(res\_mean[\#])$: raw mean residuals (for group #) (*)
- $r(res\_cov[\#])$: raw covariance residuals (for group #)
- $r(nres\_mean[\#])$: normalized mean residuals (for group #) (*)
- $r(nres\_cov[\#])$: normalized covariance residuals (for group #)
- $r(sres\_mean[\#])$: standardized mean residuals (for group #) (*)
- $r(sres\_cov[\#])$: standardized covariance residuals (for group #)

(*) If there are no estimated means or intercepts in the sem model, these matrices are not returned.

References


Also see

[SEM] sem — Structural equation model estimation command

[SEM] sem postestimation — Postestimation tools for sem

[SEM] estat eqgof — Equation-level goodness-of-fit statistics

[SEM] estat ggof — Group-level goodness-of-fit statistics

[SEM] estat gof — Goodness-of-fit statistics

[SEM] Example 10 — MIMIC model

[SEM] Methods and formulas for sem — Methods and formulas for sem
**Description**

`estat scoretests` is for use after `sem` but not `gsem`.

`estat scoretests` displays score tests (Lagrange multiplier tests) for each of the user-specified linear constraints imposed on the model when it was fit. See Sörbom (1989) and Wooldridge (2010, 421–428).

**Menu**

Statistics > SEM (structural equation modeling) > Testing and CIs > Score tests of linear constraints

**Syntax**

```
estat scoretests [, minchi2(#)]
```

**Option**

`minchi2(#)` suppresses output of tests with $\chi^2(1) < #$. By default, `estat mindices` lists values significant at the 0.05 level, corresponding to $\chi^2(1)$ value `minchi2(3.8414588)`. Specify `minchi2(0)` if you wish to see all tests.

**Remarks and examples**

See [SEM] Example 8.

**Stored results**

`estat scoretests` stores the following in `r()`:

Scalars

- `r(N_groups)` number of groups

Matrices

- `r(nobs)` sample size for each group
- `r(Cns_sctest)` matrix containing the displayed table values
References


Also see

[SEM] `sem` — Structural equation model estimation command

[SEM] `sem postestimation` — Postestimation tools for sem

[SEM] `estat ginvariant` — Tests for invariance of parameters across groups

[SEM] `estat mindices` — Modification indices

[SEM] Example 8 — Testing that coefficients are equal, and constraining them

[SEM] Methods and formulas for sem — Methods and formulas for sem
estat sd — Display variance components as standard deviations and correlations

Description

estat sd is for use after \\texttt{gsem} but not \\texttt{sem}.

\texttt{estat sd} displays the fitted variance components as standard deviations and correlations.

Menu

Statistics $\rightarrow$ SEM (structural equation modeling) $\rightarrow$ Other $\rightarrow$ Display standard deviations and correlations

Syntax

\texttt{estat sd [ , verbose post coeflegend]}

Options

\texttt{verbose} specifies that the full estimation table be displayed. By default, only the variance components are displayed. This option is implied when \texttt{post} is specified.

\texttt{post} causes \texttt{estat sd} to behave like a Stata estimation (e-class) command. \texttt{estat sd} posts the vector of calculated standard deviation and correlation parameters along with the corresponding variance–covariance matrix to \texttt{e()}, so that you can treat the estimated parameters just as you would results from any other estimation command. For example, you could use \texttt{test} to perform simultaneous tests of hypotheses on the parameters, or you could use \texttt{lincom} to create linear combinations.

The following option is not shown in the dialog box:

\texttt{coeflegend} specifies that the legend of the coefficients and how to specify them in an expression be displayed rather than displaying the statistics for the coefficients.

Remarks and examples

See [SEM] Example 31g.
Stored results

`estat sd` stores the following in `r()`:

Scalars
- `r(level)` confidence level

Matrices
- `r(b)` coefficient vector
- `r(V)` variance–covariance matrix of the estimators
- `r(table)` table of results

If `post` is specified, `estat sd` stores the following in `e()`:

Matrices
- `e(b)` coefficient vector
- `e(V)` variance–covariance matrix of the estimators

Also see

[SEM] `gsem` — Generalized structural equation model estimation command
[SEM] `gsem postestimation` — Postestimation tools for `gsem`
[SEM] `Example 31g` — Two-factor measurement model (generalized response)
Title

**estat stable — Check stability of nonrecursive system**

Description

`estat stable` is for use after `sem` but not `gsem`.

`estat stable` reports the eigenvalue stability index for nonrecursive models. The stability index is computed as the maximum modulus of the eigenvalues for the matrix of coefficients on endogenous variables predicting other endogenous variables. If the model was fit by `sem` with the `group()` option, `estat stable` reports the index for each group separately.

There are two formulas commonly used to calculate the index. `estat stable` uses the formulation of Bentler and Freeman (1983).

Menu

Statistics > SEM (structural equation modeling) > Other > Assess stability of nonrecursive systems

Syntax

`estat stable [, detail]`

Option

`detail` displays the matrix of coefficients on endogenous variables predicting other endogenous variables, also known as the $\beta$ matrix.

Remarks and examples

See *nonrecursive (structural) model (system)* in [SEM] Glossary. The issue of stability is described there. Also see Remarks and examples of [SEM] `estat teffects`.

169
Stored results

`estat stable` stores the following in r():

Scalars
- `r(N_groups)` number of groups
- `r(stindex[#])` stability index (for group #)

Matrices
- `r(nobs)` sample size for each group
- `r(Beta[#])` coefficients of endogenous variables on endogenous variables (for group #)
- `r(Re[#])` real parts of the eigenvalues of A (for group #)
- `r(Im[#])` imaginary parts of the eigenvalues of A (for group #)
- `r(Modulus[#])` modulus of the eigenvalues of A (for group #)

Reference


Also see

[SEM] `sem` — Structural equation model estimation command
[SEM] `sem postestimation` — Postestimation tools for sem
[SEM] `estat teffects` — Decomposition of effects into total, direct, and indirect
[SEM] Methods and formulas for sem — Methods and formulas for sem
estat stdize — Test standardized parameters

Description

estat stdize: is for use after sem but not gsem.
estat stdize: can be used to prefix test, lincom, testnl, and nlcom; see [SEM] test, [SEM] lincom, [SEM] testnl, and [SEM] nlcom.

These commands without a prefix work in the underlying metric of SEM, which is to say path coefficients, variances, and covariances. If the commands are prefixed with estat stdize:, they will work in the metric of standardized coefficients and correlation coefficients. There is no counterpart to variances in the standardized metric because variances are standardized to be 1.

Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Testing standardized parameters

Syntax

estat stdize: test ...
estat stdize: lincom ...
estat stdize: testnl ...
estat stdize: nlcom ...

Remarks and examples

See [SEM] Example 16.

Exercise caution when using the estat stdize: prefix to perform tests on estimated second moments (correlations). Do not test that correlations are 0. Instead, omit the estat stdize: prefix and test that covariances are 0. Covariances are more likely to be normally distributed than are correlations.

Stored results

Stored results are the results stored by the command being used with the estat stdize: prefix.
Also see

[SEM] `sem` — Structural equation model estimation command
[SEM] `sem postestimation` — Postestimation tools for sem
[SEM] `lincom` — Linear combinations of parameters
[SEM] `nlcom` — Nonlinear combinations of parameters
[SEM] `test` — Wald test of linear hypotheses
[SEM] `testnl` — Wald test of nonlinear hypotheses
[SEM] Example 16 — Correlation
[SEM] Methods and formulas for sem — Methods and formulas for sem
**Title**

| estat summarize — Report summary statistics for estimation sample |

**Description**

*estat summarize* is a standard postestimation command of Stata. This entry concerns use of *estat summarize* after *sem* or *gsem*.

*estat summarize* reports the summary statistics in the estimation sample for the observed variables in the model. *estat summarize* is mentioned here because

1. *estat summarize* cannot be used if *sem* was run on summary statistics data; see [SEM] Intro 11.
2. *estat summarize* allows the additional option *group* after estimation by *sem* and *gsem*.

**Menu**

Statistics > SEM (structural equation modeling) > Other > Estimation-sample summary statistics

**Syntax**

```
estat summarize [eqlist] [ , group estat_summ_options ]
```

**Options**

*group* may be specified if *group(varname)* was specified with *sem* or *gsem* at the time the model was fit. It requests that summary statistics be reported by group.

*estat_summ_options* are the standard options allowed by *estat summarize* and are outlined in *Options* of [R] estat summarize.

**Stored results**

See *Stored results* of [R] estat summarize.

**Also see**

[R] estat summarize — Summarize estimation sample

[SEM] gsem postestimation — Postestimation tools for gsem

[SEM] sem postestimation — Postestimation tools for sem
**estat teffects — Decomposition of effects into total, direct, and indirect**

### Description

`estat teffects` is for use after `sem` but not `gsem`.

`estat teffects` reports direct, indirect, and total effects for each path (Sobel 1987), along with standard errors obtained by the delta method.

### Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Direct and indirect effects

### Syntax

```
estat teffects [ , options ]
```

<table>
<thead>
<tr>
<th><code>options</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compact</code></td>
<td>do not display effects with no path</td>
</tr>
<tr>
<td><code>standardized</code></td>
<td>report standardized effects</td>
</tr>
<tr>
<td><code>nolabel</code></td>
<td>display group values, not labels</td>
</tr>
<tr>
<td><code>nodirect</code></td>
<td>do not display direct effects</td>
</tr>
<tr>
<td><code>noindirect</code></td>
<td>do not display indirect effects</td>
</tr>
<tr>
<td><code>nototal</code></td>
<td>do not display total effects</td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats, row spacing, and display of omitted paths</td>
</tr>
</tbody>
</table>

### Options

`compact` is a popular option. Consider the following model:

```
    . sem (y1<-y2 x1) (y2<-x2)
```

`x2` has no direct effect on `y1` but does have an indirect effect. `estat teffects` formats all its effects tables the same way by default, so there will be a row for the direct effect of `x2` on `y1` just because there is a row for the indirect effect of `x2` on `y1`. The value reported for the direct effect, of course, will be 0. `compact` says to omit these unnecessary rows.

`standardized` reports effects in standardized form, but standard errors of the standardized effects are not reported.

`nolabel` is relevant only if estimation was with `sem`’s `group()` option and the group variable has a value label. Groups are identified by group value rather than label.

`nodirect`, `noindirect`, and `nototal` suppress the display of the indicated effect. The default is to display all effects.
display_options: noci, nopvalues, noomitted, vsquish, cformat(\%fmt), pformat(\%fmt), sformat(\%fmt), and nolstretch; see [R] Estimation options. Although estat teffects is not an estimation command, it allows these options.

**Remarks and examples**

See [SEM] Example 7.

Direct effects are the path coefficients in the model.

Indirect effects are all mediating effects. For instance, consider

```
. sem ... (y1<-y2) (y1<-x2) (y2<-x3) ..., ...
```

The direct effect of \( y_2 \) on \( y_1 \) is the path coefficient \((y1<-y2)\).

In this example, changes in \( x_3 \) affect \( y_1 \). That is called the indirect effect and is the product of the path coefficients \((y2<-x3)\) and \((y1<-y2)\). If there were other paths in the model such that \( y_1 \) changed when \( x_3 \) changed, those effects would be added to the indirect effect as well. estat teffects reports total indirect effects.

The total effect is the sum of the direct and indirect effects.

When feedback loops are present in the model, such as

```
. sem ... (y1<-y2) (y1<-x2) (y2<-x3 y1) ..., ...
```

care must be taken when interpreting indirect effects. The feedback loop is when a variable indirectly affects itself, as \( y_1 \) does in the example; \( y_1 \) affects \( y_2 \) and \( y_2 \) affects \( y_1 \). Thus in calculating the indirect effect, the sum has an infinite number of terms although the term values get smaller and smaller and thus usually converge to a finite result. It is important that you check nonrecursive models for stability; see Bollen (1989, 397) and see [SEM] estat stable. Caution: if the model is unstable, the calculation of the indirect effect can sometimes still converge to a finite result.

**Stored results**

estat teffects stores the following in r():

Scalars
- \( r(N_{groups}) \) number of groups

Matrices
- \( r(nobs) \) sample size for each group
- \( r(direct) \) direct effects
- \( r(indirect) \) indirect effects
- \( r(total) \) total effects
- \( r(V_{direct}) \) covariance matrix of the direct effects
- \( r(V_{indirect}) \) covariance matrix of the indirect effects
- \( r(V_{total}) \) covariance matrix of the total effects

estat teffects with the standardized option additionally stores the following in r():

Matrices
- \( r(direct_{std}) \) standardized direct effects
- \( r(indirect_{std}) \) standardized indirect effects
- \( r(total_{std}) \) standardized total effects
References


Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM] `sem postestimation` — Postestimation tools for sem
- [SEM] `estat stable` — Check stability of nonrecursive system
- [SEM] `Methods and formulas for sem` — Methods and formulas for sem
Example 1 — Single-factor measurement model

Description

The single-factor measurement model is demonstrated using the following data:

```
use https://www.stata-press.com/data/r16/sem_1fmm
(single-factor measurement model)
.summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>500</td>
<td>99.518</td>
<td>14.35402</td>
<td>60</td>
<td>137</td>
</tr>
<tr>
<td>x2</td>
<td>500</td>
<td>99.954</td>
<td>14.1939</td>
<td>52</td>
<td>140</td>
</tr>
<tr>
<td>x3</td>
<td>500</td>
<td>99.052</td>
<td>14.26395</td>
<td>59</td>
<td>150</td>
</tr>
<tr>
<td>x4</td>
<td>500</td>
<td>94.474</td>
<td>70.11603</td>
<td>-113</td>
<td>295</td>
</tr>
</tbody>
</table>
```

.notes
_dta:
1. fictional data
2. Variables x1, x2, x3, and x4 each contain a test score designed to measure X. The test is scored to have mean 100.

See Single-factor measurement models in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Single-factor measurement model
- Satorra–Bentler scaled $\chi^2$ test
- Fitting the same model with gsem
- Fitting the same model with the Builder
- The measurement-error model interpretation

Single-factor measurement model

Below we fit the following model:

![Diagram of single-factor measurement model](image-url)
Example 1 — Single-factor measurement model

```
. sem (x1 x2 x3 x4 <- X)
Endogenous variables
Measurement:  x1 x2 x3 x4
Exogenous variables
Latent:  X
Fitting target model:
Iteration 0:  log likelihood = -8487.5905
Iteration 1:  log likelihood = -8487.2358
Iteration 2:  log likelihood = -8487.2337
Iteration 3:  log likelihood = -8487.2337
Structural equation model  Number of obs  =  500
Estimation method  =  ml
Log likelihood  =  -8487.2337
( 1)  [x1]X = 1

| OIM | Coef.  | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|-----|--------|-----------|------|------|----------------------|
| Measurement |        |           |      |      |                      |
| x1   |        |           |      |      |                      |
|      | X      | 1 (constrained) | 99.518 | 0.6412888 | 155.18 | 0.000 | 98.2611 | 100.7749 |
|      | _cons  |           |      |      |                      |
| x2   |        |           |      |      |                      |
|      | X      | 1.033249  | 0.0723898 | 14.27 | 0.000 | 0.8913676 | 1.17513   |
|      | _cons  |           |      |      |                      |
| x3   |        |           |      |      |                      |
|      | X      | 1.063876  | 0.0729725 | 14.58 | 0.000 | 0.9208526 | 1.2069    |
|      | _cons  |           |      |      |                      |
| x4   |        |           |      |      |                      |
|      | X      | 7.276754  | 0.4277638 | 17.01 | 0.000 | 6.438353 | 8.115156  |
|      | _cons  |           |      |      |                      |
| var(e.x1) |    | 115.6865  | 7.790423  | 101.3823 | 0.000 | 104.0692 | 127.3089  |
| var(e.x2) |    | 105.0445  | 7.38755   | 91.5873 | 0.000 | 94.3218 | 111.7689  |
| var(e.x3) |    | 101.2572  | 7.17635   | 88.12499 | 0.000 | 91.9688 | 107.4133  |
| var(e.x4) |    | 144.0406  | 145.2887   | 99.4838 | 0.000 | 104.0692 | 140.0692 |
| var(X) |    | 89.93921  | 11.07933   | 80.64676 | 0.000 | 71.5001 | 109.0857 |

LR test of model vs. saturated: chi2(2)  =  1.46, Prob > chi2 = 0.4827

The equations for this model are

\[
\begin{align*}
  x_1 &= \alpha_1 + X\beta_1 + e.x_1 \\
  x_2 &= \alpha_2 + X\beta_2 + e.x_2 \\
  x_3 &= \alpha_3 + X\beta_3 + e.x_3 \\
  x_4 &= \alpha_4 + X\beta_4 + e.x_4
\end{align*}
\]

Notes:

1. Variable X is latent exogenous and thus needs a normalizing constraint. The variable is anchored to the first observed variable, x1, and thus the path coefficient is constrained to be 1. See Identification 2: Normalization constraints (anchoring) in [SEM] Intro 4.
2. The path coefficients for $X \rightarrow x_1$, $X \rightarrow x_2$, and $X \rightarrow x_3$ are 1 (constrained), 1.03, and 1.06. Meanwhile, the path coefficient for $X \rightarrow x_4$ is 7.28. This is not unexpected; we at StataCorp generated these data, and the true coefficients are 1, 1, 1, and 7.

3. A test for “model versus saturated” is reported at the bottom of the output; the $\chi^2(2)$ statistic is 1.46 and its significance level is 0.4827. We cannot reject the null hypothesis of this test.

This test is a goodness-of-fit test in badness-of-fit units; a significant result implies that the model does not fit well.

More mathematically, the null hypothesis of the test is that the fitted covariance matrix and mean vector of the observed variables are equal to the matrix and vector observed in the population.

**Satorra–Bentler scaled $\chi^2$ test**

The model-versus-saturated goodness-of-fit statistic shown above does not follow the $\chi^2$ distribution that it is referred to when the data are nonnormal. Satorra and Bentler (1994) provide a scaled version of this statistic that more closely follows the mean of the reference distribution in the presence of nonnormal data. We can request this statistic and the corresponding robust standard errors by specifying the `vce(sbentler)` option.
. sem (x1 x2 x3 x4 <- X), vce(sbentler)
Endogenous variables
Measurement: x1 x2 x3 x4
Exogenous variables
Latent: X
Fitting target model:
Iteration 0: log pseudolikelihood = -8487.5905
Iteration 1: log pseudolikelihood = -8487.2358
Iteration 2: log pseudolikelihood = -8487.2337
Iteration 3: log pseudolikelihood = -8487.2337
Structural equation model  Number of obs = 500
Estimation method = ml
Log pseudolikelihood = -8487.2337
( 1) [x1]X = 1

| Satorra-Bentler          | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------------------------|-------|-----------|-------|-------|----------------------|
| Measurement              |       |           |       |       |                      |
| x1                       |       |           |       |       |                      |
|   X                      | 1     | (constrained) |       |       |                      |
|   _cons                  | 99.518| .6419311  | 155.03| 0.000 | 98.25984  100.7762 |
| x2                       |       |           |       |       |                      |
|   X                      | 1.033249| .0767608 | 13.46 | 0.000 | .8828006  1.183698 |
|   _cons                  | 99.954| .6347705  | 157.46| 0.000 | 98.70987  101.1981 |
| x3                       |       |           |       |       |                      |
|   X                      | 1.063876| .0751028 | 14.17 | 0.000 | .9166773  1.211075 |
|   _cons                  | 99.052| .6379032  | 155.28| 0.000 | 97.80173  100.3023 |
| x4                       |       |           |       |       |                      |
|   X                      | 7.276754| .4386592 | 16.59 | 0.000 | 6.416998  8.13651  |
|   _cons                  | 94.474| 3.135684  | 30.13 | 0.000 | 88.32817  100.6198 |
| var(e.x1)                | 115.6865| 7.744173  | 15.14 | 0.000 | 101.4617  131.9055 |
| var(e.x2)                | 105.0445| 6.499187  | 16.49 | 0.000 | 93.04833  118.5872 |
| var(e.x3)                | 101.2572| 7.00047   | 14.44 | 0.000 | 88.42551  115.9509 |
| var(e.x4)                | 144.0406| 145.6607  | 16.59 | 0.000 | 119.4766  105.3407 |
| var(X)                   | 89.93921| 11.2763   | 8.00  | 0.000 | 80.34416  100.5437 |

LR test of model vs. saturated: chi2(2) = 1.46, Prob > chi2 = 0.4827
Satorra-Bentler scaled test: chi2(2) = 1.59, Prob > chi2 = 0.4526

The rescaled statistic is labeled “Satorra–Bentler scaled test” and has a value of 1.59 with a significance level of 0.4526. As with the unadjusted test, we cannot reject the null hypothesis.
Fitting the same model with `gsem`

`sem` and `gsem` produce the same results for standard linear SEMs. We are going to demonstrate that just this once.

```
.gsem (x1 x2 x3 x4 <- X)
```

Fitting fixed-effects model:

- Iteration 0: log likelihood = -8948.2394
- Iteration 1: log likelihood = -8948.2394

Refining starting values:
- Grid node 0: log likelihood = -8487.5916

Fitting full model:
- Iteration 0: log likelihood = -8487.5916
- Iteration 1: log likelihood = -8487.5051
- Iteration 2: log likelihood = -8487.3836
- Iteration 3: log likelihood = -8487.2697
- Iteration 4: log likelihood = -8487.2337
- Iteration 5: log likelihood = -8487.2337

Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>Family</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>Gaussian</td>
<td>identity</td>
</tr>
<tr>
<td>x2</td>
<td>Gaussian</td>
<td>identity</td>
</tr>
<tr>
<td>x3</td>
<td>Gaussian</td>
<td>identity</td>
</tr>
<tr>
<td>x4</td>
<td>Gaussian</td>
<td>identity</td>
</tr>
</tbody>
</table>

Log likelihood = -8487.2337

(1) [x1]X = 1

|     | Coef. | Std. Err. | z    | P>|z| | 95% Conf. Interval |
|-----|-------|-----------|------|------|--------------------|
| x1  |       |           |      |      |                    |
| _X  | 1 (constrained) | 99.518 | .6412888 | 155.18 | 0.000 | 98.2611 | 100.7749 |
| _cons | 99.518 | .6412888 | 155.18 | 0.000 | 98.2611 | 100.7749 |
| x2  |       |           |      |      |                    |
| _X  | 1.033249 | .0723898 | 14.27 | 0.000 | .8913676 | 1.17513 |
| _cons | 99.954 | .6341354 | 157.62 | 0.000 | 98.71112 | 101.1969 |
| x3  |       |           |      |      |                    |
| _X  | 1.063876 | .0729725 | 14.58 | 0.000 | .9208526 | 1.206899 |
| _cons | 99.052 | .6372649 | 155.43 | 0.000 | 97.80298 | 100.301 |
| x4  |       |           |      |      |                    |
| _X  | 7.276753 | .4277636 | 17.01 | 0.000 | 6.438352 | 8.115154 |
| _cons | 94.474 | 3.132547 | 30.16 | 0.000 | 88.33432 | 100.6137 |
| var(X) | 89.93923 | 11.07933 | 70.64678 | 114.5001 |
| var(e.x1) | 115.6865 | 7.790422 | 101.3822 | 132.0089 |
| var(e.x2) | 105.0444 | 7.387549 | 91.51872 | 120.5692 |
| var(e.x3) | 101.2572 | 7.176349 | 88.12498 | 116.3463 |
| var(e.x4) | 144.0408 | 145.2886 | 9.94848 | 1040.067 |
Notes:

1. Results are virtually the same. Coefficients, variance estimates, and standard errors may differ in the last digit; for instance, $x4<-x$ was 7.276754 and now it is 7.276753. These are the kind of differences we would expect to see. *gsem* follows a different approach for obtaining results that involves far more numeric machinery, which correspondingly results in slightly less accuracy.

2. The log-likelihood values reported are the same. This model is one of the few models we could have chosen where *sem* and *gsem* would produce the same log-likelihood values. In general, *gsem* log likelihoods are on different metrics from those of *sem*. In the case where the model does not include observed exogenous variables, however, they share the same metric.

3. There is no reason to use *gsem* over *sem* when both can fit the same model. *sem* is slightly more accurate, is quicker, and has more postestimation features.

**Fitting the same model with the Builder**

Use the diagram above for reference.

1. Open the dataset.
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/sem_1fmm
   ```

2. Open a new Builder diagram.
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation*.

3. Create the measurement component for $X$.
   Select the Add measurement component tool, $\mathcal{W}$, and then click in the diagram about one-third of the way down from the top and slightly left of the center.
   In the resulting dialog box,
   a. change the *Latent variable name* to $X$;
   b. select $x1$, $x2$, $x3$, and $x4$ by using the *Measurement variables* control;
   c. select *Down* in the *Measurement direction* control;
   d. click on *OK*.
   If you wish, move the component by clicking on any variable and dragging it.
   Notice that the constraints of 1 on the paths from the error terms to the observed measures are implied, so we do not need to add these to our diagram.

4. Estimate.
   Click on the *Estimate* button, $\mathcal{E}$, in the Standard Toolbar, and then click on *OK* in the resulting *SEM estimation options* dialog box.

You can open a completed diagram in the Builder by typing
```
   . webgetsem sem_1fmm
```
The measurement-error model interpretation

As we pointed out in *Using path diagrams to specify standard linear SEMs* in [SEM] Intro 2, if we rename variable \( x_4 \) to be \( y \), we can reinterpret this measurement model as a measurement-error model. In this interpretation, \( X \) is the unobserved true value. \( x_1, x_2, \) and \( x_3 \) are each measurements of \( X \), but with error. Meanwhile, \( y \) (\( x_4 \)) is really something else entirely. Perhaps \( y \) is earnings, and we believe

\[
y = \alpha_4 + \beta_4 X + e_y
\]

We are interested in \( \beta_4 \), the effect of true \( X \) on \( y \).

If we were to go back to the data and type `regress y x1`, we would obtain an estimate of \( \beta_4 \), but we would expect that estimate to be biased toward 0 because of the errors-in-variable problem. The same applies for \( y \) on \( x_2 \) and \( y \) on \( x_3 \). If we do that, we obtain

- \( \beta_4 \) based on `regress y x1` 3.19
- \( \beta_4 \) based on `regress y x2` 3.36
- \( \beta_4 \) based on `regress y x3` 3.43

In the `sem` output above, we have an estimate of \( \beta_4 \) with the bias washed away:

\( \beta_4 \) based on `sem (y<-X)` 7.28

The number 7.28 is the value reported for (\( x_4<-X \)) in the `sem` output.

That \( \beta_4 \) might be 7.28 seems plausible because we expect that the estimate should be larger than the estimates we obtain using the variables measured with error. In fact, we can tell you that the 7.28 estimate is quite good because we at StataCorp know that the true value of \( \beta_4 \) is 7. Here is how we manufactured this fictional dataset:

```
set seed 83216
set obs 500
gen X = round(rnormal(0,10))
gen x1 = round(100 + X + rnormal(0, 10))
gen x2 = round(100 + X + rnormal(0, 10))
gen x3 = round(100 + X + rnormal(0, 10))
gen x4 = round(100 + 7*X + rnormal(0, 10))
drop X
```

The data recorded in `sem_1fmm.dta` were obviously generated using normality, the same assumption that is most often used to justify the SEM maximum likelihood estimator. In [SEM] Intro 4, we explained that the normality assumption can be relaxed and conditional normality can usually be substituted in its place.

So let’s consider nonnormal data. Let’s make \( X \) be \( \chi^2(2) \), a violently nonnormal distribution, resulting in the data-manufacturing code

```
set seed 83216
set obs 500
gen X = (rchi2(2)-2)*(10/2)
gen x1 = round(100 + X + rnormal(0, 10))
gen x2 = round(100 + X + rnormal(0, 10))
gen x3 = round(100 + X + rnormal(0, 10))
gen x4 = round(100 + 7*X + rnormal(0, 10))
drop X
```
All the `rnormal()` functions remaining in our code have to do with the assumed normality of the errors. The multiplicative and additive constants in the generation of `X` simply rescale the $\chi^2(2)$ variable to have mean 100 and standard deviation 10, which would not be important except for the subsequent `round()` functions, which themselves were unnecessary except that we wanted to produce a pretty dataset when we created the original `sem_1fmm.dta`.

In any case, if we rerun the commands with these data, we obtain

$$\beta_4 \text{ based on } \text{regress } y \text{ x1 } 3.24$$
$$\beta_4 \text{ based on } \text{regress } y \text{ x2 } 3.14$$
$$\beta_4 \text{ based on } \text{regress } y \text{ x3 } 3.36$$
$$\beta_4 \text{ based on } \text{sem (y<-x) } 7.25$$

We will not burden you with the details of running simulations to assess coverage; we will just tell you that coverage is excellent: reported test statistics and significance levels can be trusted.

By the way, errors in the variables is something that does not go away with progressively larger sample sizes. Change the code above to produce a 100,000-observation dataset instead of a 500-observation one, and you will obtain

$$\beta_4 \text{ based on } \text{regress } y \text{ x1 } 3.47$$
$$\beta_4 \text{ based on } \text{regress } y \text{ x2 } 3.48$$
$$\beta_4 \text{ based on } \text{regress } y \text{ x3 } 3.50$$
$$\beta_4 \text{ based on } \text{sem (y<-x) } 6.97$$

References


Also see

SEM sem — Structural equation model estimation command
SEM gsem — Generalized structural equation model estimation command
SEM Intro 5 — Tour of models
SEM Example 3 — Two-factor measurement model
SEM Example 24 — Reliability
SEM Example 27g — Single-factor measurement model (generalized response)
Example 2 — Creating a dataset from published covariances

Description

Williams, Eaves, and Cox (2002) publish covariances from their data. We will use those published covariances in [SEM] Example 3 to fit an SEM.

In this example, we show how to create the summary statistics dataset (SSD) that we will analyze in that example.

Remarks and examples

Remarks are presented under the following headings:

- Background
- Creating the SSD
- At this point, we could save the dataset and stop
- Labeling the SSD
- Listing the SSD

For more explanation, also see [SEM] Intro 11.

Background

In Williams, Eaves, and Cox (2002), the authors report a covariance matrix in a table that looks something like this:

<table>
<thead>
<tr>
<th></th>
<th>Affective</th>
<th>Miniscale</th>
<th>Cognitive</th>
<th>Miniscale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affective</td>
<td>1</td>
<td>2038.035</td>
<td>1631.766</td>
<td>1721.830</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1932.163</td>
<td>1688.292</td>
<td>702.969</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cognitive</td>
<td>7</td>
<td>630.518</td>
<td>500.128</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Creating the SSD

```
. clear all
. ssd init a1 a2 a3 a4 a5 c1 c2 c3 c4 c5
Summary statistics data initialized. Next use, in any order,
  ssd set observations (required)
  It is best to do this first.
  ssd set means (optional)
  Default setting is 0.
  ssd set variances or ssd set sd (optional)
  Use this only if you have set or will set correlations and, even
  then, this is optional but highly recommended. Default setting is 1.
  ssd set covariances or ssd set correlations (required)
. ssd set obs 216
(value set)
  Status:
    observations: set
    means: unset
    variances or sd: unset
    covariances or correlations: unset (required to be set)
. #delimit ;
delimiter now ;
. ssd set cov 2038.035
  > 1631.766 1932.163
  > 1393.567 1336.871 1313.809
  > 1657.299 1647.164 1273.261 2034.216
  > 1721.830 1688.292 1498.401 1677.767 2061.875
  > 659.795 702.969 585.019 656.527 775.118 630.518
  > 779.519 790.448 653.734 764.755 871.211 500.128 741.767
  >
  > 899.912 879.179 750.037 897.441 1008.287 648.935 679.970
  > 1087.409
  >
  > 775.235 739.157 659.867 751.860 895.616 550.476 603.950
  > 677.865 855.272
  >
  > 821.170 785.419 669.951 802.825 877.681 491.042 574.775
  > 686.391 622.830 728.674 ;
(values set)
  Status:
    observations: set
    means: unset
    variances or sd: set
    covariances or correlations: set
. #delimit cr
delimiter now cr
```

Notes:

1. We used `#delimit` to temporarily set the end-of-line character to semicolon. That was not necessary, but it made it easier to enter the data in a way that would be subsequently more readable. You can use `#delimit` only in do-files; see `[P] #delimit`.
2. We recommend entering SSD by using do-files. That way, you can edit the file and get it right.
3. We did not have to reset the delimiter. We could have entered the numbers on one (long) line. That works well when there are only a few summary statistics.
At this point, we could save the dataset and stop

We could save the dataset and stop right now if we wished:

```
. save sem_2fmm
file sem_2fmm.dta saved
```

Obviously, we can save the dataset anytime we wish. We know we could stop because `ssd status` tells us whether there is anything more that we need to define:

```
. ssd status
Status:
    observations: set
    means: unset
    variances or sd: set
    covariances or correlations: set
```

Notes:
1. The means have not been set. The authors did not provide the means.
2. `ssd status` would mention if anything that was not set was required to be set.

Labeling the SSD

If we were to use `ssd describe` to describe these data, the output would look like this:

```
. ssd describe
Summary statistics data
    obs: 216
    vars: 10

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td></td>
</tr>
<tr>
<td>a2</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td></td>
</tr>
<tr>
<td>a5</td>
<td></td>
</tr>
<tr>
<td>c1</td>
<td></td>
</tr>
<tr>
<td>c2</td>
<td></td>
</tr>
<tr>
<td>c3</td>
<td></td>
</tr>
<tr>
<td>c4</td>
<td></td>
</tr>
<tr>
<td>c5</td>
<td></td>
</tr>
</tbody>
</table>
```

We can add labels and notes to our dataset:

```
. label data "Affective and cognitive arousal"
. label var a1 "affective arousal 1"
. label var a2 "affective arousal 2"
. label var a3 "affective arousal 3"
. label var a4 "affective arousal 4"
. label var a5 "affective arousal 5"
. label var c1 "cognitive arousal 1"
. label var c2 "cognitive arousal 2"
. label var c3 "cognitive arousal 3"
. label var c4 "cognitive arousal 4"
```
Example 2 — Creating a dataset from published covariances

. label var c5 "cognitive arousal 5"
. #delimit ;
delimiter now ;
. notes: Summary statistics data containing published covariances
  > from Thomas O. Williams, Ronald C. Eaves, and Cynthia Cox,
  > 2 Apr 2002, "Confirmatory factor analysis of an instrument
design to measure affective and cognitive arousal",
  > _Educational and Psychological Measurement_,
  > vol. 62 no. 2, 264-283. ;
. notes: a1-a5 report scores from 5 miniscales designed to measure
  > affective arousal. ;
. notes: c1-c5 report scores from 5 miniscales designed to measure
cognitive arousal. ;
. notes: The series of tests, known as the VST II
  > (Visual Similes Test II) were administered to 216 children
  > ages 10 to 12. The miniscales are sums of scores of
  > 5 to 6 items in VST II. ;
. #delimit cr
delimiter now cr
. ssd describe
Summary statistics data
  obs: 216 Affective and cognitive arousal
  vars: 10 (_dta has notes)

variable name        variable label
a1      affective arousal 1
a2      affective arousal 2
a3      affective arousal 3
a4      affective arousal 4
a5      affective arousal 5
c1      cognitive arousal 1
c2      cognitive arousal 2
c3      cognitive arousal 3
c4      cognitive arousal 4
c5      cognitive arousal 5

. save sem_2fmm, replace
file sem_2fmm.dta saved

Notes:
1. You can label the variables and the data, and you can add notes just as you would to any dataset.
2. You save and use SSD just as you save and use any dataset.
### Listing the SSD

```
.ssd list
Observations = 216
Means undefined; assumed to be 0
Variances implicitly defined; they are the diagonal of the covariance matrix.
Covariances:
   a1   a2   a3   a4   a5   c1   c2
2038.035
1631.766 1932.163
1393.567 1336.871 1313.809
1657.299 1647.164 1273.261 2034.216
1721.83 1688.292 1498.401 1677.767 2061.875
659.795 702.969 585.019 656.527 775.118 630.518
779.519 790.448 653.734 764.755 871.211 500.128 741.767
899.912 879.179 750.037 897.441 1008.287 648.935 679.97
775.235 739.157 659.867 751.86 895.616 550.476 603.95
821.17 785.419 669.951 802.825 877.681 491.042 574.775
c3   c4   c5
1087.409
677.865 855.272
686.391 622.83 728.674
```

### Reference


### Also see

[SEM] **Example 3** — Two-factor measurement model

[SEM] **ssd** — Making summary statistics data (sem only)
Example 3 — Two-factor measurement model

Description

The multiple-factor measurement model is demonstrated using summary statistics dataset (SSD) `sem_2fmm.dta`:

```
. use https://www.stata-press.com/data/r16/sem_2fmm
   (Affective and cognitive arousal)
. ssd describe
```

Summary statistics data from https://www.stata-press.com/data/r16/sem_2fmm.dta

```
obs: 216  Affective and cognitive arousal
vars: 10  25 May 2018 10:11
(_dta has notes)
```

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>affective arousal 1</td>
</tr>
<tr>
<td>a2</td>
<td>affective arousal 2</td>
</tr>
<tr>
<td>a3</td>
<td>affective arousal 3</td>
</tr>
<tr>
<td>a4</td>
<td>affective arousal 4</td>
</tr>
<tr>
<td>a5</td>
<td>affective arousal 5</td>
</tr>
<tr>
<td>c1</td>
<td>cognitive arousal 1</td>
</tr>
<tr>
<td>c2</td>
<td>cognitive arousal 2</td>
</tr>
<tr>
<td>c3</td>
<td>cognitive arousal 3</td>
</tr>
<tr>
<td>c4</td>
<td>cognitive arousal 4</td>
</tr>
<tr>
<td>c5</td>
<td>cognitive arousal 5</td>
</tr>
</tbody>
</table>

```
. notes
_dta:
1. Summary statistics data containing published covariances from Thomas O.
   Williams, Ronald C. Eaves, and Cynthia Cox, 2 Apr 2002, “Confirmatory
   factor analysis of an instrument designed to measure affective and
   cognitive arousal”, _Educational and Psychological Measurement_, vol. 62
   no. 2, 264-283.
2. a1-a5 report scores from 5 miniscales designed to measure affective
   arousal.
3. c1-c5 report scores from 5 miniscales designed to measure cognitive
   arousal.
4. The series of tests, known as the VST II (Visual Similes Test II) were
   administered to 216 children ages 10 to 12. The miniscales are sums of
   scores of 5 to 6 items in VST II.
```

See [SEM] Example 2 to learn how we created this SSD.
Remarks and examples

Remarks are presented under the following headings:

- Fitting multiple-factor measurement models
- Displaying standardized results
- Fitting the model with the Builder
- Obtaining equation-level goodness of fit by using estat eqgof

See *Multiple-factor measurement models* in [SEM] Intro 5 for background.

Fitting multiple-factor measurement models

Below we fit the model shown by Kline (2005, 70–74, 184), namely,

```
. sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5)
```

Endogenous variables
Measurement: a1 a2 a3 a4 a5 c1 c2 c3 c4 c5
Exogenous variables
Latent: Affective Cognitive

Fitting target model:
Iteration 0: log likelihood = -9542.8803
Iteration 1: log likelihood = -9539.5505
Iteration 2: log likelihood = -9539.3856
Iteration 3: log likelihood = -9539.3851
Example 3 — Two-factor measurement model

| Measurement | DIN | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------|-----|-------|-----------|-----|------|---------------------|
| a1 Affective |     | 1 (constrained) |       |     |      |                     |
| a2 Affective |     | .9758098 | .0460752  | 21.18 | 0.000 | .885504 1.066116     |
| a3 Affective |     | .8372599 | .0355086  | 23.58 | 0.000 | .767664  .9068556    |
| a4 Affective |     | .9640461 | .0499203  | 19.31 | 0.000 | .866204 1.061888     |
| a5 Affective |     | 1.063701 | .0435751  | 24.41 | 0.000 | .978295 1.149107     |
| c1 Cognitive |     | 1 (constrained) |       |     |      |                     |
| c2 Cognitive |     | 1.114702 | .0655687  | 17.00 | 0.000 | .986190 1.243215     |
| c3 Cognitive |     | 1.329882 | .0791968  | 16.79 | 0.000 | 1.174659 1.485105    |
| c4 Cognitive |     | 1.172792 | .0711692  | 16.48 | 0.000 | 1.033303 1.312281    |
| c5 Cognitive |     | 1.126356 | .0644475  | 17.48 | 0.000 | 1.000041 1.252671    |
| var(e.a1)   |     | 384.1359 | 43.79119  |      |       | 400.3095           |
| var(e.a2)   |     | 357.3524 | 41.00499  |      |       | 447.4755           |
| var(e.a3)   |     | 154.9507 | 20.9026   |      |       | 199.7822           |
| var(e.a4)   |     | 496.4594 | 54.16323  |      |       | 614.8214           |
| var(e.a5)   |     | 191.6857 | 28.07212  |      |       | 255.4154           |
| var(e.c1)   |     | 171.6638 | 19.82327  |      |       | 215.2649           |
| var(e.c2)   |     | 171.8055 | 20.53479  |      |       | 217.1579           |
| var(e.c3)   |     | 276.0144 | 32.33535  |      |       | 347.2569           |
| var(e.c4)   |     | 224.1994 | 25.93412  |      |       | 281.2527           |
| var(e.c5)   |     | 146.8655 | 18.5756   |      |       | 188.1829           |
| var(Affect-e) |   | 164.463  | 193.1032  |      |       | 2070.034           |
| var(Cognit-e)| | 455.9349 | 59.11245  |      |       | 587.8439           |
| cov(Affect-e, Cognitive) | | 702.0736 | 85.72272  | 8.19 | 0.000 | 534.0601 870.087    |

LR test of model vs. saturated: chi2(34) = 88.88, Prob > chi2 = 0.0000
Notes:

1. In [SEM] Example 1, we ran sem on raw data. In this example, we run sem on SSD. There are no special sem options that we need to specify because of this.

2. The estimated coefficients reported above are unstandardized coefficients or, if you prefer, factor loadings.

3. The coefficients listed at the bottom of the coefficient table that start with e, are the estimated error variances. They represent the variance of the indicated measurement that is not measured by the respective latent variables.

4. The above results do not match exactly (Kline 2005, 184). If we specified sem option nm1, results are more likely to match to 3 or 4 digits. The nm1 option says to divide by \( N - 1 \) rather than by \( N \) in producing variances and covariances.

Displaying standardized results

The output will be easier to interpret if we display standardized values for paths rather than path coefficients. A standardized value is in standard deviation units. It is the change in one variable given a change in another, both measured in standard deviation units. We can obtain standardized values by specifying sem’s standardized option, which we can do when we fit the model or when we replay results:
Example 3 — Two-factor measurement model

```
. sem, standardized
Structural equation model
  Number of obs = 216
  Estimation method = ml
  Log likelihood = -9539.3851

  ( 1) [a1]Affective = 1
  ( 2) [c1]Cognitive = 1

| Standardized Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------------------|-----------|------|------|---------------------|
| Measurement       |           |      |      |                     |
| a1 Affective      | .9003553  | .0143988 | 62.53 | .000             | .8721342 .9285765 |
| a2 Affective      | .9023249  | .0141867 | 63.60 | .000             | .8745195 .9301304 |
| a3 Affective      | .9388883  | .0097501 | 96.29 | .000             | .9197784 .9579983 |
| a4 Affective      | .8687982  | .0181922 | 47.76 | .000             | .8331421 .9044543 |
| a5 Affective      | .9521559  | .0083489 | 114.05 | .000            | .9357923 .9685195 |
| c1 Cognitive      | .8523351  | .0212439 | 40.12 | .000             | .8106978 .8939725 |
| c2 Cognitive      | .8759601  | .0184216 | 47.55 | .000             | .8398544 .9120658 |
| c3 Cognitive      | .863129   | .0199624 | 43.24 | .000             | .8240033 .9022547 |
| c4 Cognitive      | .8582786  | .0204477 | 41.97 | .000             | .8182018 .8983554 |
| c5 Cognitive      | .8930346  | .0166261 | 53.71 | .000             | .8604479 .9256212 |
| var(e.a1)         | .1893602  | .0259281 |      | .1447899 .2476506 |
| var(e.a2)         | .1858097  | .0256021 |      | .1418353 .2434179 |
| var(e.a3)         | .1184887  | .0183086 |      | .0875289 .1603993 |
| var(e.a4)         | .2451896  | .0316107 |      | .1904417 .3156764 |
| var(e.a5)         | .0933991  | .015899  |      | .0669031 .1303885 |
| var(e.c1)         | .2735248  | .0362139 |      | .2110086 .354563 |
| var(e.c2)         | .2326939  | .0322732 |      | .1773081 .3053806 |
| var(e.c3)         | .2550083  | .0344603 |      | .1956717 .3323885 |
| var(e.c4)         | .2633578  | .0350997 |      | .2028151 .3419733 |
| var(e.c5)         | .2024893  | .0296954 |      | .1519049 .2699183 |
| var(Affect~e)     | 1         |       |      | .            |
| var(Cognitive~e)  | 1         |       |      | .            |
| cov(Affect~e, Cognitive) | .8108102 | .0268853 | 30.16 | .000     | .758116 .8635045 |

LR test of model vs. saturated: chi2(34) = 88.88, Prob > chi2 = 0.0000
```
Notes:

1. In addition to obtaining standardized coefficients, the **standardized** option reports estimated error variances as the fraction of the variance that is unexplained. Error variances were previously unintelligible numbers such as 384.136 and 357.352. Now they are 0.189 and 0.186.

2. Also listed in the **sem** output are variances of latent variables. In the previous output, latent variable *Affective* had variance 1,644.46 with standard error 193. In the standardized output, it has variance 1 with standard error missing. The variances of the latent variables are standardized to 1, and obviously, being a normalization, there is no corresponding standard error.

3. We can now see at the bottom of the coefficient table that affective and cognitive arousal are correlated 0.81 because standardized covariances are correlation coefficients.

4. The standardized coefficients for this model can be interpreted as the correlation coefficients between the indicator and the latent variable because each indicator measures only one factor. For instance, the standardized path coefficient \( a1 \leftarrow \text{Affective} \) is 0.90, meaning the correlation between \( a1 \) and *Affective* is 0.90.

### Fitting the model with the Builder

Use the diagram above for reference.

1. Open the dataset.
   - In the Command window, type 
     ```
     . use https://www.stata-press.com/data/r16/sem_2fmm
     ```

2. Open a new Builder diagram.
   - Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Change the size of the observed variables’ rectangles.
   - From the **SEM Builder** menu, select **Settings > Variables > All observed...**.
   - In the resulting dialog box, change the first size to 0.38 and click on **OK**.

4. Create the measurement component for affective arousal.
   - Select the Add measurement component tool, ⧬, and then click in the diagram about one-third of the way down from the top and one-fourth of the way in from the left.
   - In the resulting dialog box,
     a. change the **Latent variable name** to *Affective*;
     b. select \( a1 \), \( a2 \), \( a3 \), \( a4 \), and \( a5 \) by using the **Measurement variables** control;
     c. select **Down** in the **Measurement direction** control;
     d. click on **OK**.
   - If you wish, move this component by clicking on any variable and dragging it.

5. Create the measurement component for cognitive arousal.
   - Repeat the process from item 4, but place the measurement component about one-third of the way down from the top and three-fourths of the way in from the left. Label the latent variable *Cognitive*, and select measurement variables \( c1 \), \( c2 \), \( c3 \), \( c4 \), and \( c5 \). Drag to reposition if desired.
6. Correlate the latent factors.
   a. Select the Add covariance tool, \( \rightleftharpoons \).
   b. Click in the upper-right quadrant of the Affective oval (it will highlight when you hover over it), and drag a covariance to the upper-left quadrant of the Cognitive oval (it will highlight when you can release to connect the covariance).

7. Clean up.
   If you do not like where a covariance has been connected to its variable, use the Select tool, \( \rightleftharpoons \), to click on the covariance, and then simply click on where it connects to an oval and drag the endpoint. You can also change the bow of the covariance by dragging the control point that extends from one end of the selected covariance.

8. Estimate.
   Click on the Estimate button, \( \rightleftharpoons \), in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

9. Show standardized estimates.
   From the SEM Builder menu, select View > Standardized estimates.

   You can open a completed diagram in the Builder by typing
   \[ . \ \text{webgetsem} \ \text{sem}_2\text{fmm} \]

### Obtaining equation-level goodness of fit by using estat eqgof

That the correlation between \( a_1 \) and Affective is 0.90 implies that the fraction of the variance of \( a_1 \) explained by Affective is \( 0.90^2 = 0.81 \), and left unexplained is \( 1 - 0.81 = 0.19 \). Instead of manually calculating the proportion of variance explained by indicators, we can use the estat eqgof command:

\[ . \ \text{estat eqgof} \]

**Equation-level goodness of fit**

<table>
<thead>
<tr>
<th>depvars</th>
<th>Variance</th>
<th>R-squared</th>
<th>mc</th>
<th>mc2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fitted</td>
<td>predicted</td>
<td>residual</td>
<td></td>
</tr>
<tr>
<td>observed</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a1</td>
<td>2028.598</td>
<td>1644.463</td>
<td>384.1359</td>
<td>.8106398</td>
</tr>
<tr>
<td>a2</td>
<td>1923.217</td>
<td>1565.865</td>
<td>357.3524</td>
<td>.8141903</td>
</tr>
<tr>
<td>a3</td>
<td>1307.726</td>
<td>1152.775</td>
<td>154.9507</td>
<td>.8815113</td>
</tr>
<tr>
<td>a4</td>
<td>2024.798</td>
<td>1528.339</td>
<td>496.4594</td>
<td>.7548104</td>
</tr>
<tr>
<td>a5</td>
<td>2052.328</td>
<td>1860.643</td>
<td>191.6857</td>
<td>.9066009</td>
</tr>
<tr>
<td>c1</td>
<td>627.5987</td>
<td>455.9349</td>
<td>171.6638</td>
<td>.7264752</td>
</tr>
<tr>
<td>c2</td>
<td>738.3325</td>
<td>566.527</td>
<td>171.8055</td>
<td>.7673061</td>
</tr>
<tr>
<td>c3</td>
<td>1082.374</td>
<td>806.3598</td>
<td>276.0144</td>
<td>.7449917</td>
</tr>
<tr>
<td>c4</td>
<td>851.311</td>
<td>627.1116</td>
<td>224.1994</td>
<td>.7366422</td>
</tr>
<tr>
<td>c5</td>
<td>725.3002</td>
<td>578.4346</td>
<td>146.8655</td>
<td>.7975107</td>
</tr>
<tr>
<td>overall</td>
<td></td>
<td></td>
<td></td>
<td>.9949997</td>
</tr>
</tbody>
</table>

\( mc \) = correlation between depvar and its prediction
\( mc^2 \) = \( mc^2 \) is the Bentler-Raykov squared multiple correlation coefficient
Notes:

1. **fitted** reports the fitted variance of each of the endogenous variables, whether observed or latent. In this case, we have observed endogenous variables.

2. **predicted** reports the variance of the predicted value of each endogenous variable.

3. **residual** reports the leftover residual variance.

4. **R-squared** reports $R^2$, the fraction of variance explained by each indicator. The fraction of the variance of Affective explained by $a_1$ is 0.81, just as we calculated by hand at the beginning of this section. The overall $R^2$ is also called the coefficient of determination.

5. **mc** stands for multiple correlation, and **mc2** stands for multiple-correlation squared. **R-squared**, **mc**, and **mc2** all report the relatedness of the indicated dependent variable with the model’s linear prediction. In recursive models, all three statistics are really the same number. mc is equal to the square root of R-squared, and mc2 is equal to R-squared.

   In nonrecursive models, these three statistics are different and each can have problems. R-squared and mc can actually become negative! That does not mean the model has negative predictive power or that it might not even have reasonable predictive power. mc2 = mc^2 is recommended by Bentler and Raykov (2000) to be used instead of R-squared for nonrecursive systems.

In [SEM] **Example 4**, we examine the goodness-of-fit statistics for this model.

In [SEM] **Example 5**, we examine modification indices for this model.

References


Also see

[SEM] **Example 1** — Single-factor measurement model

[SEM] **Example 2** — Creating a dataset from published covariances

[SEM] **Example 20** — Two-factor measurement model by group

[SEM] **Example 26** — Fitting a model with data missing at random

[SEM] **Example 31g** — Two-factor measurement model (generalized response)

[SEM] **sem** — Structural equation model estimation command

[SEM] **estat eqgof** — Equation-level goodness-of-fit statistics
Example 4 — Goodness-of-fit statistics

Description

Here we demonstrate estat gof. See [SEM] Intro 7 and [SEM] estat gof.
This example picks up where [SEM] Example 3 left off:

```
. use https://www.stata-press.com/data/r16/sem_2fmm
. sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5)
```

Remarks and examples

When we fit this model in [SEM] Example 3, at the bottom of the output, we saw

```
. sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5)
(output omitted)
LR test of model vs. saturated: chi2(34) = 88.88, Prob > chi2 = 0.0000
```

Most texts refer to this test against the saturated model as the “model χ² test”.

These results indicate poor goodness of fit; see [SEM] Example 1. The default goodness-of-fit statistic reported by sem, however, can be overly influenced by sample size, correlations, variance unrelated to the model, and multivariate nonnormality (Kline 2016, 271).

Goodness of fit in cases of sem is a measure of how well you fit the observed moments, which in this case are the covariances between all pairs of a1, ..., a5, c1, ..., c5. In a measurement model, the assumed underlying causes are unobserved, and in this example, those unobserved causes are the latent variables Affective and Cognitive. It may be reasonable to assume that the observed a1, ..., a5, c1, ..., c5 can be filtered through imagined variables Affective and Cognitive, but that can be reasonable only if not too much information contained in the original variables is lost. Thus goodness-of-fit statistics are of great interest to those fitting measurement models. Goodness-of-fit statistics are of far less interest when all variables in the model are observed.
Other goodness-of-fit statistics are available.

. estat gof, stats(all)

<table>
<thead>
<tr>
<th>Fit statistic</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chi2_ms(34)</td>
<td>88.879</td>
<td>model vs. saturated</td>
</tr>
<tr>
<td>p &gt; chi2</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>chi2_bs(45)</td>
<td>2467.161</td>
<td>baseline vs. saturated</td>
</tr>
<tr>
<td>p &gt; chi2</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>Population error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RMSEA</td>
<td>0.087</td>
<td>Root mean squared error of approximation</td>
</tr>
<tr>
<td>90% CI, lower bound</td>
<td>0.065</td>
<td></td>
</tr>
<tr>
<td>upper bound</td>
<td>0.109</td>
<td></td>
</tr>
<tr>
<td>pclose</td>
<td>0.004</td>
<td>Probability RMSEA &lt;= 0.05</td>
</tr>
<tr>
<td>Information criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIC</td>
<td>19120.770</td>
<td>Akaike’s information criterion</td>
</tr>
<tr>
<td>BIC</td>
<td>19191.651</td>
<td>Bayesian information criterion</td>
</tr>
<tr>
<td>Baseline comparison</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CFI</td>
<td>0.977</td>
<td>Comparative fit index</td>
</tr>
<tr>
<td>TLI</td>
<td>0.970</td>
<td>Tucker-Lewis index</td>
</tr>
<tr>
<td>Size of residuals</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRMR</td>
<td>0.022</td>
<td>Standardized root mean squared residual</td>
</tr>
<tr>
<td>CD</td>
<td>0.995</td>
<td>Coefficient of determination</td>
</tr>
</tbody>
</table>

Notes:

1. Desirable values vary from test to test.
2. We asked for all the goodness-of-fit tests. We could have obtained specific tests from the above output by specifying the appropriate option; see [SEM] estat gof.
3. Under likelihood ratio, estat gof reports two tests. The first is a repeat of the model $\chi^2$ test reported at the bottom of the sem output. The saturated model is the model that fits the covariances perfectly. We can reject at the 5% level (or any other level) that the model fits as well as the saturated model.

The second test is a baseline versus saturated comparison. The baseline model includes the mean and variances of all observed variables plus the covariances of all observed exogenous variables. Different authors define the baseline differently. We can reject at the 5% level (or any other level) that the baseline model fits as well as the saturated model.

4. Under population error, the RMSEA value is reported along with the lower and upper bounds of its 90% confidence interval. Most interpreters of this test check whether the lower bound is below 0.05 or the upper bound is above 0.10. If the lower bound is below 0.05, then they would not reject the hypothesis that the fit is close. If the upper bound is above 0.10, they would not reject the hypothesis that the fit is poor. The logic is to perform one test on each end of the 90% confidence interval and thus have 95% confidence in the result. This model’s fit is not close, and its upper limit is just over the bounds of being considered poor.

Pclose, a commonly used word in reference to this test, is the probability that the RMSEA value is less than 0.05, interpreted as the probability that the predicted moments are close to the moments in the population. This model’s fit is not close.
5. Under information criteria are reported AIC and BIC, which contain little information by themselves but are often used to compare models. Smaller values are considered better.

6. Under baseline comparison are reported CFI and TLI, two indices such that a value close to 1 indicates a good fit. TLI is also known as the nonnormed fit index.

7. Under size of residuals is reported the standardized root mean squared residual (SRMR) and the coefficient of determination (CD).

   A perfect fit corresponds to an SRMR of 0, and a good fit corresponds to a “small” value, considered by some to be limited at 0.08. The model fits well by this standard.

   The CD is like an $R^2$ for the whole model. A value close to 1 indicates a good fit.

`estat gof` provides multiple goodness-of-fit statistics because, across fields, different researchers use different statistics. You should not print them all and look for the one reporting the result you seek.

Reference


Also see

[SEM] Example 3 — Two-factor measurement model
[SEM] Example 21 — Group-level goodness of fit
[SEM] `estat gof` — Goodness-of-fit statistics
Example 5 — Modification indices

Description

Here we demonstrate the use of estat mindices; see [SEM] Intro 7 and [SEM] estat mindices. This example picks up where [SEM] Example 3 left off:

```
use https://www.stata-press.com/data/r16/sem_2fmm
sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5)
```

Remarks and examples

When we fit this model in [SEM] Example 4, we allowed the latent variables to be correlated. We typed

```
sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5)
```

and by default in the command language, latent exogenous variables are assumed to be correlated unless we specify otherwise. Had we used the Builder, the latent exogenous variables would have been assumed to be uncorrelated unless we had drawn the curved path between them.

The original authors who collected these data analyzed them assuming no covariance, which we could obtain by typing

```
sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5), ///
cov(Affective*Cognitive@0)
```

It was Kline (2005, 70–74, 184) who allowed the covariance. Possibly he did that after looking at the modification indices.

The modification indices report statistics on all omitted paths and covariances. Let’s begin with the model without the covariance:
. sem (Affective -> a1 a2 a3 a4 a5) (Cognitive -> c1 c2 c3 c4 c5),
> cov(Affective*Cognitive@0)
(output omitted)
. estat mindices

<table>
<thead>
<tr>
<th>Modification indices</th>
<th>MI</th>
<th>df</th>
<th>P&gt;MI</th>
<th>EPC</th>
<th>Standard EPC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cognitive</td>
<td>8.059</td>
<td>1</td>
<td>0.00</td>
<td>.1604476</td>
<td>.075774</td>
</tr>
<tr>
<td>c5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Affective</td>
<td>5.885</td>
<td>1</td>
<td>0.02</td>
<td>.0580897</td>
<td>.087733</td>
</tr>
<tr>
<td>cov(e.a1,e.a4)</td>
<td>5.767</td>
<td>1</td>
<td>0.02</td>
<td>84.81133</td>
<td>.1972802</td>
</tr>
<tr>
<td>cov(e.a1,e.a5)</td>
<td>7.597</td>
<td>1</td>
<td>0.01</td>
<td>-81.82092</td>
<td>-.2938627</td>
</tr>
<tr>
<td>cov(e.a2,e.a4)</td>
<td>14.300</td>
<td>1</td>
<td>0.00</td>
<td>129.761</td>
<td>.3110565</td>
</tr>
<tr>
<td>cov(e.a2,e.c4)</td>
<td>4.071</td>
<td>1</td>
<td>0.04</td>
<td>-45.44807</td>
<td>-.1641344</td>
</tr>
<tr>
<td>cov(e.a3,e.a4)</td>
<td>21.183</td>
<td>1</td>
<td>0.00</td>
<td>-116.8181</td>
<td>-.4267012</td>
</tr>
<tr>
<td>cov(e.a3,e.a5)</td>
<td>25.232</td>
<td>1</td>
<td>0.00</td>
<td>118.4674</td>
<td>.6681337</td>
</tr>
<tr>
<td>cov(e.a5,e.c4)</td>
<td>4.209</td>
<td>1</td>
<td>0.04</td>
<td>39.07999</td>
<td>.184049</td>
</tr>
<tr>
<td>cov(e.c1,e.c3)</td>
<td>11.326</td>
<td>1</td>
<td>0.00</td>
<td>66.3965</td>
<td>.3098331</td>
</tr>
<tr>
<td>cov(e.c1,e.c5)</td>
<td>8.984</td>
<td>1</td>
<td>0.00</td>
<td>-47.31483</td>
<td>-.2931597</td>
</tr>
<tr>
<td>cov(e.c3,e.c4)</td>
<td>12.668</td>
<td>1</td>
<td>0.00</td>
<td>-80.98353</td>
<td>-.333871</td>
</tr>
<tr>
<td>cov(e.c4,e.c5)</td>
<td>4.483</td>
<td>1</td>
<td>0.03</td>
<td>38.6556</td>
<td>.2116015</td>
</tr>
<tr>
<td>cov(Affective,Cognitive)</td>
<td>128.482</td>
<td>1</td>
<td>0.00</td>
<td>704.4469</td>
<td>.8094959</td>
</tr>
</tbody>
</table>

EPC = expected parameter change

Notes:

1. Four columns of results are reported.
   a. MI stands for modification index and is an approximation to the change in the model’s
      goodness-of-fit $\chi^2$ if the path were added.
   b. df stands for degrees of freedom and is the number that would be added to $d$ of the $\chi^2(d)$.
   c. P>MI is the value of the significance of $\chi^2(df)$.
   d. EPC stands for expected parameter change and is an approximation to the value of the
      parameter if it were not constrained to 0. It is reported in unstandardized (column 3) and
      standardized (column 4) units.

2. There are lots of significant omitted paths and covariances in the above output.

3. Paths and covariances are listed only if the modification index is significant at the 0.05 level,
   corresponding to $\chi^2(1)$ value 3.8414588. You may specify the minchi2() option to use different
   $\chi^2(1)$ values. Specify minchi2(0) if you wish to see all tests.

4. The omitted covariance between Affective and Cognitive has the largest change in $\chi^2$
   observed. Perhaps this is why Kline (2005, 70–74, 184) allowed a covariance between the two
   latent variables. The standardized EPC reports the relaxed-constraint correlation value, which is
   the value reported for the unconstrained correlation path in [SEM] Example 3.

Another way of dealing with this significant result would be to add a direct path between the
variables, but that perhaps would have invalidated the theory being proposed. The original authors
instead proposed a second-order model postulating that Affective and Cognitive are themselves
measurements of another latent variable that might be called Arousal.
Reference


Also see

[SEM] Example 3 — Two-factor measurement model
[SEM] estat mindices — Modification indices
Linear regression is demonstrated using auto.dta:

```
. sysuse auto
(1978 Automobile Data)
```

See *Structural models 1: Linear regression* in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting linear regression models
- Displaying standardized results
- Fitting the model with the Builder

Fitting linear regression models

The first two examples in [R] regress are

```
. regress mpg weight c.weight#c.weight foreign
. regress, beta
```

This model corresponds to

![Diagram](image-url)
Example 6 — Linear regression 205

To fit this model with `sem`, we type

```
. generate weight2 = weight^2
. sem (mpg <- weight weight2 foreign)
```

Endogenous variables
Observed: mpg

Exogenous variables
Observed: weight weight2 foreign

Fitting target model:
Iteration 0:  log likelihood = -1909.8206
Iteration 1:  log likelihood = -1909.8206

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef. Std. Err. z P&gt;</td>
</tr>
<tr>
<td>Structural</td>
<td></td>
</tr>
<tr>
<td>mpg</td>
<td>56.53884 6.027559 9.38  0.000  44.72504 68.35264</td>
</tr>
<tr>
<td>weight</td>
<td>-0.1656729 .0038604 -4.29  0.000 -.0241392 -.0090067</td>
</tr>
<tr>
<td>weight2</td>
<td>1.59e-06 6.08e-07 2.62  0.009  4.00e-07 2.78e-06</td>
</tr>
<tr>
<td>foreign</td>
<td>-2.2035 1.03022  -2.14  0.032 -4.222695 -.1843056</td>
</tr>
<tr>
<td>_cons</td>
<td>56.53884 6.027559 9.38  0.000  44.72504 68.35264</td>
</tr>
</tbody>
</table>

Note: The LR test of model vs. saturated is not reported because the fitted model is not full rank.

Notes:

1. We wished to include variable `weight^2` in our model. Because `sem` does not allow Stata’s factor-variable notation, we first had to generate new variable `weight2`.

2. Reported coefficients match those reported by `regress`.

3. Reported standard errors (SEs) differ slightly from those reported by `regress`. For instance, the SE for `foreign` is reported here as 1.03, whereas `regress` reported 1.06. `SEM` is an asymptotic estimator, and `sem` divides variances and covariances by \( N = 74 \), the number of observations. `regress` provides unbiased finite-sample estimates and divides by \( N - k - 1 = 74 - 3 - 1 = 70 \). Note that \( 1.03 \sqrt{74/70} = 1.06 \).

4. `sem` reports \( z \) statistics whereas `regress` reports \( t \) statistics.

5. Reported confidence intervals differ slightly between `sem` and `regress` because of the \( (N - k - 1)/N \) issue.

6. `sem` reports the point estimate of \( e.mpg \) as 10.19332. `regress` reports the root MSE as 3.2827. And \( \sqrt{10.19332 \times 74/70} = 3.2827 \).

Displaying standardized results

To obtain standardized coefficients from `regress`, you specify the `beta` option. To obtain standardized coefficients from `sem`, you specify the `standardized` option.
Example 6 — Linear regression

```
> . sem, standardized
Structural equation model                Number of obs =  74
Estimation method = ml                   Log likelihood = -1909.8206

          |      OIM
          |     Coef.  Std. Err.     z    P>|z|    [95% Conf. Interval]
Structural
  mpg
    weight         -2.226321    .4950378   -4.50  0.000   -3.196577   -1.256064
    weight2        1.326541    .4982614    2.66  0.008    .3499662   2.303113
    foreign       -.1752702    .0810378   -2.16  0.031   -.3341011   -.0164389
    _cons          9.839209    .9686872   10.16  0.000    7.940617   11.7378

  var(e.mpg)    .3087043    .0482719    6.41  0.000    .2272168   .4194152

Note: The LR test of model vs. saturated is not reported because the fitted model is not full rank.
```

`regress` simply reports standardized coefficients in an extra column. All other results are reported in unstandardized form. `sem` updates the entire output with the standardized values.

**Fitting the model with the Builder**

Use the diagram above for reference.

1. Open the dataset and create the additional variable `weight2`.
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/auto
   . generate weight2 = weight^2
   ```

2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Create the regression component for the `mpg` outcome.
   Select the Add regression component tool, and then click in the center of the diagram.
   In the resulting dialog box,
   a. select `mpg` in the Dependent variable control;
   b. select `weight`, `weight2`, and `foreign` by using the Independent variables control;
   c. select Left in the Independent variables’ direction control;
   d. click on OK.
   If you wish, move this component by clicking on any variable and dragging it.

4. Estimate.
   Click on the Estimate button, in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

5. Show standardized estimates.
   From the SEM Builder menu, select View > Standardized estimates.
You can open a completed diagram in the Builder by typing

```
.webgetsem sem_regress
```

Also see

- **[SEM] Example 12** — Seemingly unrelated regression
- **[SEM] Example 38g** — Random-intercept and random-slope models (multilevel)
- **[SEM] Example 43g** — Tobit regression
- **[SEM] Example 44g** — Interval regression
- **[SEM] sem** — Structural equation model estimation command
Example 7 — Nonrecursive structural model

Description

To demonstrate a nonrecursive structural model with all variables observed, we use data from Duncan, Haller, and Portes (1968):

```
. use https://www.stata-press.com/data/r16/sem_sm1
(Structural model with all observed values)
. ssd describe
```

Summary statistics data from
https://www.stata-press.com/data/r16/sem_sm1.dta

```
obs: 329 Structural model with all observed values
vars: 10 25 May 2018 10:13
(_dta has notes)
```

<table>
<thead>
<tr>
<th>variable</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_intel</td>
<td>respondent’s intelligence</td>
</tr>
<tr>
<td>r_parasp</td>
<td>respondent’s parental aspiration</td>
</tr>
<tr>
<td>r_ses</td>
<td>respondent’s family socioeconomic status</td>
</tr>
<tr>
<td>r_occasp</td>
<td>respondent’s occupational aspiration</td>
</tr>
<tr>
<td>r_educaasp</td>
<td>respondent’s educational aspiration</td>
</tr>
<tr>
<td>f_intel</td>
<td>friend’s intelligence</td>
</tr>
<tr>
<td>f_parasp</td>
<td>friend’s parental aspiration</td>
</tr>
<tr>
<td>f_ses</td>
<td>friend’s family socioeconomic status</td>
</tr>
<tr>
<td>f_occasp</td>
<td>friend’s occupational aspiration</td>
</tr>
<tr>
<td>f_educaasp</td>
<td>friend’s educational aspiration</td>
</tr>
</tbody>
</table>

```
. notes
_dta:
  2. The data contain 329 boys with information on five variables and the same information for each boy’s best friend.

If you typed `ssd status`, you would learn that this dataset contains the correlation matrix only. Variances (standard deviations) and means are undefined. Thus we need to use this dataset cautiously. It is always better if you enter the variances and means if you have them.

That these data are the correlations only will not matter for how we will use them.
Remarks and examples

See *Structural models 8: Dependencies between response variables* in [SEM] Intro 5 for background. Remarks are presented under the following headings:

- Fitting the model
- Fitting the model with the Builder
- Checking stability with estat stable
- Reporting total, direct, and indirect effects with estat teffects

Fitting the model

In the referenced paper above, the authors fit the following model:
. sem (r_occasp <- f_occasp r_intel r_ses f_ses)
> (f_occasp <- r_occasp f_intel f_ses r_ses),
> cov(e.r_occasp*e.f_occasp) standardized

Endogenous variables

Observed: r_occasp f_occasp

Exogenous variables

Observed: r_intel r_ses f_ses f_intel

Fitting target model:

Iteration 0: log likelihood = -2617.0489
Iteration 1: log likelihood = -2617.0489

Structural equation model

Number of obs = 329

Estimation method = ml
Log likelihood = -2617.0489

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td><strong>Structural</strong></td>
<td></td>
</tr>
<tr>
<td>r_occasp</td>
<td></td>
</tr>
<tr>
<td>f_occasp</td>
<td>.2773441</td>
</tr>
<tr>
<td>r_intel</td>
<td>.2854766</td>
</tr>
<tr>
<td>r_ses</td>
<td>.1570082</td>
</tr>
<tr>
<td>f_ses</td>
<td>.0973327</td>
</tr>
<tr>
<td>f_occasp</td>
<td></td>
</tr>
<tr>
<td>r_occasp</td>
<td>.2118102</td>
</tr>
<tr>
<td>r_ses</td>
<td>.0794194</td>
</tr>
<tr>
<td>f_ses</td>
<td>.1681772</td>
</tr>
<tr>
<td>f_intel</td>
<td>.3693682</td>
</tr>
<tr>
<td>var(e.r_occasp)</td>
<td>.6889244</td>
</tr>
<tr>
<td>var(e.f_occasp)</td>
<td>.6378539</td>
</tr>
<tr>
<td>cov(e.r_occasp,</td>
<td>-.2325666</td>
</tr>
<tr>
<td>e.f_occasp)</td>
<td></td>
</tr>
</tbody>
</table>

LR test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .

Notes:

1. We specified the standardized option, but in this case that did not matter much because these data are based on the correlation coefficients only, so standardized values are equal to unstandardized values. The exception is the correlation between the latent endogenous variables, as reflected in the correlation of their errors, and we wanted to show that results match those in the original paper.

2. Nearly all results match those in the original paper. The authors normalized the errors to have a variance of 1; sem normalizes the paths from the errors to have coefficient 1. While you can apply most normalizing constraints any way you wish, sem restricts errors to have path coefficients of 1 and this cannot be modified. You could, however, prove to yourself that sem would produce the same variances as the authors produced by typing

```
. sem, coeflegend
. display sqrt(_b[/var(e.r_occasp)])
. display sqrt(_b[/var(e.f_occasp)])
```
because the coefficients would be the standard deviations of the errors estimated without the variance-1 constraint. Thus all results match. We replayed results by using the `coeflegend` option so that we would know what to type to refer to the two error variances, namely, `b[/var(e.r_occasp)]` and `b[/var(e.f_occasp)]`.

**Fitting the model with the Builder**

Use the diagram above for reference.

1. Open the dataset.
   
   In the Command window, type
   
   ```
   . use https://www.stata-press.com/data/r16/sem_sm1
   ```

2. Open a new Builder diagram.
   
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Create the four independent variables.
   
   Select the Add observed variables set tool, and then click in the diagram about one-third of the way in from the left and one-fourth of the way up from the bottom.
   
   In the resulting dialog box,
   
   a. select the **Select variables** radio button (it may already be selected);
   
   b. use the **Variables** control to select the four variables in this order: `r_intel`, `r_ses`, `f_ses`, and `f_intel`;
   
   c. select **Vertical** in the **Orientation** control;
   
   d. click on **OK**.
   
   If you wish, move the set of variables by clicking on any variable and dragging it.

4. Create the two dependent variables.
   
   Select the Add observed variables set tool, and then click in the diagram about two-thirds of the way in from the left, vertically aligned with the top of the `f_intel` rectangle.
   
   In the resulting dialog box,
   
   a. select the **Select variables** radio button (it may already be selected);
   
   b. use the **Variables** control to select the two variable names `r_occasp` and `f_occasp`;
   
   c. select **Vertical** in the **Orientation** control;
   
   d. select the **Distances** tab;
   
   e. select `.5` (inch) from the **Distance between variables** control;
   
   f. click on **OK**.
   
   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create paths from the independent variables to the dependent variables.
   
   a. Select the Add path tool, `-`.
   
   b. Click in the right side of the `r_intel` rectangle (it will highlight when you hover over it), and drag a path to the left side of the `r_occasp` rectangle (it will highlight when you can release to connect the path).
c. Continuing with the tool, create the following paths by first clicking in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable.

\[
\begin{align*}
\text{r_ses} & \rightarrow \text{r_occasp} \\
\text{r_ses} & \rightarrow \text{f_occasp} \\
\text{f_ses} & \rightarrow \text{r_occasp} \\
\text{f_ses} & \rightarrow \text{f_occasp} \\
\text{f_intel} & \rightarrow \text{f_occasp}
\end{align*}
\]

6. Create paths for the feedback loop between endogenous variables.

a. Continue with the Add path tool.

b. Click in the bottom of the \text{r_occasp} rectangle, slightly to the left of the center, and drag a path to the \text{f_occasp} rectangle.

c. Click in the top of the \text{f_occasp} rectangle, slightly to the right of the center, and drag a path to the \text{r_occasp} rectangle.

7. Correlate the error terms.

a. Select the Add covariance tool.

b. Click in the error term for \text{r_occasp} (the circle labeled $\epsilon_1$), and drag a covariance to the error term for \text{f_occasp} (the circle labeled $\epsilon_2$).

8. Clean up.

If you do not like where a path has been connected to its variable, use the Select tool, to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint. Similarly, you can change where the covariance connects to the error terms by clicking on the covariance and dragging the endpoint. You can also change the bow of the covariance by clicking on the covariance and dragging the control point that extends from one end of the selected covariance.


Click on the Estimate button, in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

10. Show standardized estimates.

From the SEM Builder menu, select View > Standardized estimates.

Tips: When you draw paths that should be exactly horizontal or vertical, such as the two between \text{r_occasp} and \text{f_occasp}, holding the Shift key as you drag the path will guarantee that the line is perfectly vertical. Also, when drawing paths from the independent variables to the dependent variables, you may find it more convenient to change the automation settings as described in the tips of [SEM] Example 9. However, this does not work for the feedback loop between the dependent variables.

You can open a completed diagram in the Builder by typing

.webgetsem sem_sm1
Checking stability with `estat stable`

```stata
. estat stable
Stability analysis of simultaneous equation systems
   Eigenvalue stability condition

<table>
<thead>
<tr>
<th>Eigenvalue</th>
<th>Modulus</th>
</tr>
</thead>
<tbody>
<tr>
<td>.2423722</td>
<td>.242372</td>
</tr>
<tr>
<td>-.2423722</td>
<td>.242372</td>
</tr>
</tbody>
</table>

stability index = .2423722
All the eigenvalues lie inside the unit circle.
SEM satisfies stability condition.
```

Notes:

1. `estat stable` is for use on nonrecursive models. Recursive models are, by design, stable.
2. Stability concerns whether the parameters of the model are such that the model would blow up if it were operated over and over again. If the results are found not to be stable, then that casts questions about the validity of the model.
3. The stability is the maximum of the moduli, and the moduli are the absolute values of the eigenvalues. Usually, the two eigenvalues are not identical, but it is a property of this model that they are.
4. If the stability index is less than 1, then the reported estimates yield a stable model.

In the next section, we use `estat teffects` to estimate total effects. That is appropriate only if the model is stable, as we find that it is.
## Reporting total, direct, and indirect effects with *estat teffects*

```
* estat teffects

### Direct effects

| OIM                  | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------------|-------|-----------|-------|-----|-----------------------|
| Structural           |       |           |       |     |                       |
| r_occasp             |       |           |       |     |                       |
| r_occasp             | 0     | (no path) |       |     |                       |
| f_occasp             | .2773441 | .1287622 | 2.15  | 0.031 | .0249748 -.5297134   |
| r_intel              | .2854766 | .0522001 | 5.47  | 0.000 | .1831662 .3877869   |
| r_ses                | .1570082 | .052733  | 2.98  | 0.003 | .0536534 .260363   |
| f_ses                | .0973327 | .0603699 | 1.61  | 0.107 | -0.0209901 .2156555 |
| f_intel              | 0     | (no path) |       |     |                       |
| f_occasp             | .2118102 | .1563958 | 1.35  | 0.176 | -.09472 .5183404   |
| f_occasp             | 0     | (no path) |       |     |                       |
| r_intel              | .0794194 | .0589095 | 1.35  | 0.178 | -.0360411 .1948799 |
| r_ses                | .1681772 | .0543854 | 3.09  | 0.002 | .0615838 .2747705 |
| f_ses                | .3693682 | .0557939 | 6.62  | 0.000 | .2600142 .4787223 |

### Indirect effects

| OIM                  | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------------|-------|-----------|-------|-----|-----------------------|
| Structural           |       |           |       |     |                       |
| r_occasp             |       |           |       |     |                       |
| r_occasp             | .0624106 | .0594905 | 1.05  | 0.294 | -.0541886 .1790097 |
| f_occasp             | .0173092 | .0220932 | 0.78  | 0.433 | -.0259928 .0606112 |
| r_intel              | .0178168 | .0159383 | 1.12  | 0.264 | -.0134217 .0490552 |
| r_ses                | .0332001 | .0204531 | 1.62  | 0.105 | -.0068872 .0732875 |
| f_ses                | .0556285 | .0292043 | 1.90  | 0.057 | -.0016109 .112868 |
| f_intel              | .1088356 | .052243  | 2.08  | 0.037 | .0064411 .21123   |
| f_occasp             | .0132192 | .0215569 | 0.61  | 0.540 | -.0390316 .05547 |
| f_occasp             | .0624106 | .0594905 | 1.05  | 0.294 | -.0541886 .1790097 |
| r_intel              | .0642406 | .0490164 | 1.31  | 0.190 | -.0318298 .160311 |
| r_ses                | .0402881 | .0315496 | 1.28  | 0.202 | -.021548 .1021242 |
| f_ses                | .0323987 | .0262124 | 1.24  | 0.216 | -.0189765 .083774 |
| f_intel              | .0230525 | .0202112 | 1.14  | 0.254 | -.0165607 .0626657 |
```
Total effects

| QIM | Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-----|-------|-----------|-------|-------|---------------------|
| Structural |       |           |       |       |                     |
| r_occasp | .0624106 | .0594905 | 1.05  | 0.294 | -.0541886 .1790097 |
| f_occasp | .2946533 | .1468389 | 2.01  | 0.045 | .0068544 .5824522  |
| r_intel  | .3032933 | .0509684 | 5.95  | 0.000 | .2033971 .4031896  |
| r_ses    | .1902083 | .050319  | 3.78  | 0.000 | .091585 .2888317  |
| f_ses    | .1529612 | .050844  | 3.01  | 0.003 | .0533089 .2526136  |
| f_intel  | .1088356 | .052243  | 2.08  | 0.037 | .0064411 .21123   |

Note:

1. In the path diagram we drew for this model, you can see that the intelligence of the respondent, \( r_{\text{intel}} \), has both direct and indirect effects on the occupational aspiration of the respondent, \( r_{\text{occasp}} \). The tables above reveal that

\[
0.3033 = 0.2855 + 0.0178
\]

where 0.2855 is the direct effect and 0.0178 is the indirect effect.

References


Also see

- [SEM] Example 8 — Testing that coefficients are equal, and constraining them
- [SEM] *sem* — Structural equation model estimation command
- [SEM] *estat stable* — Check stability of nonrecursive system
- [SEM] *estat teffects* — Decomposition of effects into total, direct, and indirect
Example 8 — Testing that coefficients are equal, and constraining them

**Description**

This example continues where [SEM] Example 7 left off, where we typed

```
. use https://www.stata-press.com/data/r16/sem_sm1
. ssd describe
. notes
. sem (r_occasp <- f_occasp r_intel r_ses f_ses) ///
   (f_occasp <- r_occasp f_intel f_ses r_ses), ///
   cov(e.r_occasp*e.f_occasp) standardized
. estat stable
. estat teffects
```

**Remarks and examples**

Remarks are presented under the following headings:

- **Using test to evaluate adding constraints**
- **Refitting the model with added constraints**
- **Using estat scoretests to test whether constraints can be relaxed**

We want to show you how to evaluate potential constraints after estimation, how to fit a model with constraints, and how to evaluate enforced constraints after estimation.

Obviously, in a real analysis, if you evaluated potential constraints after estimation, there would be no reason to evaluate enforced constraints after estimation, and vice versa.

**Using test to evaluate adding constraints**

In this model of respondents and corresponding friends, it would be surprising if the coefficients relating friends’ characteristics to respondents’ occupational aspirations and vice versa were not equal. It would also be surprising if coefficients relating a respondent’s characteristics to his occupational aspirations were not equal to those of his friends’ characteristics to his occupational aspirations. The paths that we suspect should be equal are

- `r_intel -> r_occasp`
- `f_intel -> f_occasp`
- `r_ses -> r_occasp`
- `f_ses -> r_occasp`
- `f_ses -> r_occasp`
- `r_ses -> f_occasp`
- `r_occasp -> r_occasp`
- `r_occasp -> f_occasp`

You are about to learn that to test whether those paths have equal coefficients, you type

```
. test (_b[r_occasp:r_intel ]==_b[f_occasp:f_intel ])
   ///
   (_b[r_occasp:r_ses ]==_b[f_occasp:f_ses ])
   ///
   (_b[r_occasp:f_ses ]==_b[f_occasp:r_ses ])
   ///
   (_b[r_occasp:f_occasp]==_b[f_occasp:r_occasp])
```

216
In Stata, \_b[] is how one accesses the estimated parameters. It is difficult to remember what the names are. To determine the names of the parameters, replay the `sem` results with the `coeflegend` option:

```
. sem, coeflegend
```

```
Structural equation model

Estimation method = ml
Log likelihood = -2617.0489

Number of obs = 329
```

```

<table>
<thead>
<tr>
<th>Coef. Legend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structural</td>
</tr>
<tr>
<td>r_occasp</td>
</tr>
<tr>
<td>f_occasp</td>
</tr>
<tr>
<td>r_intel</td>
</tr>
<tr>
<td>r_ses</td>
</tr>
<tr>
<td>f_ses</td>
</tr>
</tbody>
</table>

| f_occasp     |
| r_occasp     | .2118102 \_b[f_occasp:r_occasp] |
| r_ses        | .0794194 \_b[f_occasp:r_ses] |
| f_ses        | .1681772 \_b[f_occasp:f_ses] |
| f_intel      | .3693682 \_b[f_occasp:f_intel] |

| var(e.r_occ~p) | .6868304 \_b[/var(e.r_occasp)] |
| var(e.f_occ~p) | .6359151 \_b[/var(e.f_occasp)] |

| cov(e.r_occ~p,|
| e.f_occasp)  | -.1536992 \_b[/cov(e.r_occasp,e.f_occasp)] |

LR test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .
```

With the parameter names at hand, to perform the test, we can type

```
. test (_b[r_occasp:r_intel ]==_b[f_occasp:f_intel ])
>   ( 1) [r_occasp]r_intel - [f_occasp]f_intel = 0
>   ( 2) [r_occasp]r_ses - [f_occasp]f_ses = 0
>   ( 3) [r_occasp]f_ses - [f_occasp]r_ses = 0
>   ( 4) [r_occasp]f_occasp - [f_occasp]r_occasp = 0

   chi2( 4) = 1.61
   Prob > chi2 = 0.8062
```

We cannot reject the constraint, just as we expected.
Refitting the model with added constraints

We could refit the model with these constraints by typing

```
    . sem (r_occasp <- f_occasp @ b1 r_intel @ b2 r_ses @ b3 f_ses @ b4)
    > (f_occasp <- r_occasp @ b1 f_intel @ b2 f_ses @ b3 r_ses @ b4),
    >     cov(e.r_occasp*e.f_occasp)
```

Endogenous variables
Observed: r_occasp f_occasp

Exogenous variables
Observed: r_intel r_ses f_ses f_intel

Fitting target model:
Iteration 0: log likelihood = -2617.8735
Iteration 1: log likelihood = -2617.8705
Iteration 2: log likelihood = -2617.8705

Structural equation model

| Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------|-----------|-------|------|----------------------|
| r_occasp | f_occasp | .2471578 | .1024504 | 2.41 | 0.016 | .0463588 | .4479568 |
| r_intel | r_occasp | .3271847 | .0407973 | 8.02 | 0.000 | .2472234 | .4071459 |
| r_ses   | r_occasp | .1635056 | .0380582 | 4.30 | 0.000 | .0889129 | .2380984 |
| f_ses   | r_occasp | .088364  | .0427106 | 2.07 | 0.039 | .0046529 | .1720752 |
|         | f_occasp | .2471578 | .1024504 | 2.41 | 0.016 | .0463588 | .4479568 |
|         | r_ses   | .088364  | .0427106 | 2.07 | 0.039 | .0046529 | .1720752 |
|         | f_ses   | .1635056 | .0380582 | 4.30 | 0.000 | .0889129 | .2380984 |
|         | f_intel | .3271847 | .0407973 | 8.02 | 0.000 | .2472234 | .4071459 |
| var(e.r_occasp) | .6884513 | .0538641 | .5905757 | .8025477 |
| var(e.f_occasp) | .6364713 | .0496867 | .5461715 | .7417005 |
| cov(e.r_occasp, e.f_occasp) | -.1582175 | .1410111 | -.4345942 | .1181592 |

LR test of model vs. saturated: chi2(4) = 1.64, Prob > chi2 = 0.8010

Using estat scoretests to test whether constraints can be relaxed

```
    . estat scoretests
```

(no score tests to report; all chi2 values less than 3.841458820694123)
No tests were reported because no tests were individually significant at the 5% level. We can obtain all the individual tests by adding the minchi2(0) option, which we can abbreviate to min(0):

```
estat scoretests, min(0)
```

Score tests for linear constraints

<table>
<thead>
<tr>
<th></th>
<th>chi2</th>
<th>df</th>
<th>P&gt;chi2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1) [r_occasp]f_occasp - [f_occasp]r_occasp = 0</td>
<td>0.014</td>
<td>1</td>
<td>0.91</td>
</tr>
<tr>
<td>( 2) [r_occasp]r_intel - [f_occasp]f_intel = 0</td>
<td>1.225</td>
<td>1</td>
<td>0.27</td>
</tr>
<tr>
<td>( 3) [r_occasp]r_ses - [f_occasp]f_ses = 0</td>
<td>0.055</td>
<td>1</td>
<td>0.81</td>
</tr>
<tr>
<td>( 4) [r_occasp]f_ses - [f_occasp]r_ses = 0</td>
<td>0.136</td>
<td>1</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Notes:

1. When we began this example, we used `test` to evaluate potential constraints that we were considering. We obtained an overall $\chi^2(4)$ statistic of 1.61 and thus could not reject the constraints at any reasonable level.
2. We then refit the model with those constraints.
3. For pedantic reasons, now we use `estat scoretests` to evaluate relaxing constraints included in the model. `estat scoretests` does not report a joint test. You cannot sum the $\chi^2$ values to obtain a joint test statistic. Thus we learn only that the individual constraints should not be relaxed at reasonable confidence levels.
4. Thus when evaluating multiple constraints, it is better to fit the model without the constraints and use `test` to evaluate them jointly.

**Also see**

[SEM] Example 7 — Nonrecursive structural model

[SEM] `sem` — Structural equation model estimation command

[SEM] `sem and gsem path notation` — Command syntax for path diagrams

[SEM] `estat scoretests` — Score tests

[SEM] `test` — Wald test of linear hypotheses
Example 9 — Structural model with measurement component

Description

To demonstrate a structural model with a measurement component, we use data from Wheaton et al. (1977):

```
. use https://www.stata-press.com/data/r16/sem_sm2
(Structural model with measurement component)

. ssd describe
Summary statistics data from https://www.stata-press.com/data/r16/sem_sm2.dta
obs: 932 Structural model with measurement component
vars: 13 25 May 2018 11:45
(_dta has notes)

variable name     variable label                  
----------------- ----------------------------
educ66            Education, 1966
occstat66         Occupational status, 1966
anomia66          Anomia, 1966
pwless66          Powerlessness, 1966
socdist66         Latin American social distance, 1966
occstat67         Occupational status, 1967
anomia67          Anomia, 1967
pwless67          Powerlessness, 1967
socdist67         Latin American social distance, 1967
occstat71         Occupational status, 1971
anomia71          Anomia, 1971
pwless71          Powerlessness, 1971
socdist71         Latin American social distance, 1971
```

. notes
_dta:

See Structural models 9: Unobserved inputs, outputs, or both in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

  Fitting the model
  Fitting the model with the Builder
  Evaluating omitted paths with estat mindices
  Refitting the model
Fitting the model

Simplified versions of the model fit by the authors of the referenced paper appear in many SEM software manuals. A simplified model is

```
. sem
> (anomia67 pwless67 <- Alien67) /// measurement piece
> (anomia71 pwless71 <- Alien71) /// measurement piece
> (Alien67 <- SES) /// structural piece
> (Alien71 <- Alien67 SES) /// structural piece
> ( SES -> educ66 occstat66) // measurement piece

Endogenous variables
Measurement: anomia67 pwless67 anomia71 pwless71 educ66 occstat66
Latent: Alien67 Alien71

Exogenous variables
Latent: SES

Fitting target model:
Iteration 0: log likelihood = -15249.988
Iteration 1: log likelihood = -15246.584
Iteration 2: log likelihood = -15246.469
Iteration 3: log likelihood = -15246.469
```
### Example 9 — Structural model with measurement component

Structural equation model  
Number of obs = 932  
Estimation method = ml  
Log likelihood = -15246.469

( 1) [anomia67]Alien67 = 1  
( 2) [anomia71]Alien71 = 1  
( 3) [educ66]SES = 1

|             | Coef.     | Std. Err. | z         | P>|z|       | [95% Conf. Interval] |
|-------------|-----------|-----------|-----------|-----------|---------------------|
| Structural  |           |           |           |           |                     |
| Alien67     | -.6140404 | .0562407  | -10.92    | 0.000     | -.7242701 -.5038107 |
| SES         | .7046342  | .0533512  | 13.21     | 0.000     | .6000678 .8092007   |
| Alien71     | -.1744153 | .0542489  | -3.22     | 0.001     | -.2807413 -.0680894 |
| SES         |           |           |           |           |                     |

| Measurement |           |           |           |           |                     |
| anomia67    |           |           |           |           |                     |
| Alien67     | .8884887  | .0431565  | 20.59     | 0.000     | .8039034 .9730739   |
| _cons       | 13.61     | .1126205  | 120.85    | 0.000     | 13.38927 13.83073   |
| pwless67    |           |           |           |           |                     |
| Alien67     | .8486022  | .0415205  | 20.44     | 0.000     | .7672235 .9299808   |
| _cons       | 14.9      | .1034537  | 144.03    | 0.000     | 14.69723 15.10277   |
| anomia71    |           |           |           |           |                     |
| Alien71     | 14.13     | .1158943  | 121.92    | 0.000     | 13.90285 14.35715   |
| _cons       |           |           |           |           |                     |
| pwless71    |           |           |           |           |                     |
| Alien71     | .8092007  | .0533512  | 13.21     | 0.000     | .6000678 .8092007   |
| _cons       | .7046342  | .0533512  | 13.21     | 0.000     | .6000678 .8092007   |
| educ66      |           |           |           |           |                     |
| SES         | 10.9      | .1014894  | 107.40    | 0.000     | 10.70108 11.09892   |
| _cons       |           |           |           |           |                     |
| occstat66   |           |           |           |           |                     |
| SES         | 35.49     | .6947112  | 53.96     | 0.000     | 35.12839 38.85161   |
| _cons       |           |           |           |           |                     |
| var(e.ano~67)| 4.009921  | .3582978  | 11.32     | 0.000     | 3.356724 4.777416   |
| var(e.pwl~67)| 3.187468  | .283374   | 11.56     | 0.000     | 2.677762 3.794197   |
| var(e.ano~71)| 3.695593  | .3911512  | 9.43      | 0.000     | 3.003245 4.54755    |
| var(e.pwl~71)| 3.621531  | .3037908  | 12.02     | 0.000     | 3.072483 4.268693   |
| var(e.educ66)| 2.943819  | .5002527  | 5.88      | 0.000     | 2.109908 4.107319   |
| var(e.occ~66)| 260.63    | 18.24572  | 14.34     | 0.000     | 227.2139 298.9605   |
| var(e.Ali~67)| 5.301416  | .483144   | 10.95     | 0.000     | 4.344225 6.338201   |
| var(e.Ali~71)| 3.737286  | .3881546  | 9.68      | 0.000     | 3.048951 4.581019   |
| var(SES)    | 6.65587   | .6409484  | 10.61     | 0.000     | 5.511067 8.038482   |

LR test of model vs. saturated: chi2(6) = 71.62, Prob > chi2 = 0.0000

Notes:

1. Measurement component: In both 1967 and 1971, anomia and powerlessness are used to measure endogenous latent variables representing alienation for the same two years. Education and occupational status are used to measure the exogenous latent variable SES.

3. The model versus saturated $\chi^2$ test indicates that the model is a poor fit.

**Fitting the model with the Builder**

Use the diagram above for reference.

1. Open the dataset.
   
   In the Command window, type
   
   . use https://www.stata-press.com/data/r16/sem_sm2

2. Open a new Builder diagram.
   
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation.**

3. Create the measurement component for alienation in 1967.
   
   Select the Add measurement component tool, $\mathcal{C}$, and then click in the diagram about halfway down from the top and one-third of the way in from the left.
   
   In the resulting dialog box,
   
   a. change the **Latent variable name** to Alien67;
   
   b. select anomia67 and pwless67 by using the **Measurement variables** control;
   
   c. select **Up** in the **Measurement direction** control;
   
   d. click on **OK.**
   
   If you wish, move the component by clicking on any variable and dragging it.

4. Create the measurement component for alienation in 1971.
   
   Repeat the process from item 3, but place the measurement component about halfway down from the top and two-thirds of the way in from the left. Label the latent variable Alien71, and select measurement variables anomia71 and pwless71. Again drag to reposition if desired.

5. Create the measurement component for socioeconomic status.
   
   Select the Add measurement component tool, $\mathcal{C}$, and then click in the diagram about one-third of the way up from the bottom and one-fourth of the way in from the left.
   
   In the resulting dialog box,
   
   a. change the **Latent variable name** to SES;
   
   b. select educ66 and occstat66 by using the **Measurement variables** control;
   
   c. select **Down** in the **Measurement direction** control;
   
   d. click on **OK.**

6. Create a path from Alien67 to Alien71.
   
   a. Select the Add path tool, $\rightarrow$.
   
   b. Click in the right half of the Alien67 oval (it will highlight when you hover over it), and drag a path to the left half of the Alien71 oval (it will highlight when you can release to connect the path).
7. Create a path from SES to Alien67.
   Continuing with the tool, click in the upper portion of the SES oval, and drag a path to the lower-left quadrant of the Alien67 oval.

8. Create a path from SES to Alien71.
   Continuing with the tool, click in the upper-right quadrant of the SES oval, and drag a path to the lower-left quadrant of the Alien71 oval.

9. Clean up the direction of the errors.
   The error on Alien67 may have been created under the path from Alien67 to Alien71. If so, choose the Select tool, , and then click in the Alien67 oval. Click on one of the Error rotation buttons, , in the Contextual Toolbar until the error is where you want it.

10. Estimate.
    Click on the Estimate button, , in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

Tips: We took some care above to click in specific quadrants of the ovals when creating paths. This is because, by default, the Builder places the ends of paths on the boundary of the ovals and rectangles closest to where we click. When you create paths, it is often convenient to change the automation settings (Settings > Automation...) so that Connection attachment points is Attach based on position of variables rather than Attach nearest to drop point. With the Connection attachment points setting, the paths connect nicely regardless of where we click or drop in the variables’ ovals and rectangles.

Note that when you create covariances with the Covariances tool, , Attach nearest to drop point is sometimes preferred.

You can open a completed diagram in the Builder by typing

    . webgetsem sem_sm2
Evaluating omitted paths with estat mindices

That the model is a poor fit leads us to look at the modification indices:

```
. estat mindices
```

Modification indices

<table>
<thead>
<tr>
<th>Measurement</th>
<th>MI</th>
<th>df</th>
<th>P&gt;MI</th>
<th>EPC</th>
<th>EPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>anomia67</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>anomia71</td>
<td>51.977</td>
<td>1</td>
<td>0.00</td>
<td>0.3906425</td>
<td>0.4019984</td>
</tr>
<tr>
<td>pwless71</td>
<td>32.517</td>
<td>1</td>
<td>0.00</td>
<td>-0.2969297</td>
<td>-0.2727609</td>
</tr>
<tr>
<td>educ66</td>
<td>5.627</td>
<td>1</td>
<td>0.02</td>
<td>0.0935048</td>
<td>0.0842631</td>
</tr>
<tr>
<td>pwless67</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>anomia71</td>
<td>41.618</td>
<td>1</td>
<td>0.00</td>
<td>-0.3106995</td>
<td>-0.3594367</td>
</tr>
<tr>
<td>pwless71</td>
<td>23.622</td>
<td>1</td>
<td>0.00</td>
<td>0.2249714</td>
<td>0.2323233</td>
</tr>
<tr>
<td>educ66</td>
<td>6.441</td>
<td>1</td>
<td>0.01</td>
<td>-0.0889042</td>
<td>-0.0900664</td>
</tr>
<tr>
<td>anomia71</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>anomia67</td>
<td>58.768</td>
<td>1</td>
<td>0.00</td>
<td>0.429437</td>
<td>0.4173061</td>
</tr>
<tr>
<td>pwless71</td>
<td>38.142</td>
<td>1</td>
<td>0.00</td>
<td>-0.3873066</td>
<td>-0.3347904</td>
</tr>
<tr>
<td>pwless67</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>anomia71</td>
<td>46.188</td>
<td>1</td>
<td>0.00</td>
<td>-0.3308484</td>
<td>-0.3601641</td>
</tr>
<tr>
<td>pwless71</td>
<td>27.760</td>
<td>1</td>
<td>0.00</td>
<td>0.2871709</td>
<td>0.2780833</td>
</tr>
<tr>
<td>educ66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>anomia67</td>
<td>4.415</td>
<td>1</td>
<td>0.04</td>
<td>0.1055965</td>
<td>0.1171781</td>
</tr>
<tr>
<td>pwless71</td>
<td>6.816</td>
<td>1</td>
<td>0.01</td>
<td>-0.1469371</td>
<td>-0.1450411</td>
</tr>
</tbody>
</table>

EPC = expected parameter change

Notes:

1. There are lots of statistically significant paths we could add to the model.
2. Some of those statistically significant paths also make theoretical sense.
3. Two in particular that make theoretical sense are the covariances between `e.anomia67` and `e.anomia71` and between `e.pwless67` and `e.pwless71`. 
Refitting the model

Let’s refit the model and include those two previously excluded covariances:

```
. sem ///
> (anomia67 pwless67 <- Alien67) /// measurement piece
> (anomia71 pwless71 <- Alien71) /// measurement piece
> (Alien67 <- SES) /// structural piece
> (Alien71 <- Alien67 SES) /// structural piece
> ( SES -> educ66 occstat66) /// measurement piece
> , cov(e.anomia67*e.anomia71) ///
>   cov(e.pwless67*e.pwless71)
```

Endogenous variables
Measurement: anomia67 pwless67 anomia71 pwless71 educ66 occstat66
Latent: Alien67 Alien71

Exogenous variables
Latent: SES

Fitting target model:
Iteration 0:  log likelihood = -15249.988
Iteration 1:  log likelihood = -15217.95
Iteration 2:  log likelihood = -15213.126
Iteration 3:  log likelihood = -15213.046
Iteration 4:  log likelihood = -15213.046
### Example 9 — Structural model with measurement component

#### Structural equation model

- **Number of obs:** 932
- **Estimation method:** ml
- **Log likelihood:** -15213.046

#### Structural

| Coef.       | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-------------|-----------|-------|-------|----------------------|
| Alien67     | -.5752228 | .057961 | -9.92 | 0.000                | -.6888244 , -.4616213 |
| SES         | .606954   | .0512305 | 11.85 | 0.000                | .5065439 , .707364    |
| Alien71     | -.2270301 | .0530773 | -4.28 | 0.000                | -.3310596 , -.1230006 |
| SES         | -.2270301 | .0530773 | -4.28 | 0.000                | -.3310596 , -.1230006 |

#### Measurement

| Coef.       | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-------------|-----------|-------|-------|----------------------|
| anomia67    | 1 (constrained) | 13.61 | .1126143 | 120.85 | 0.000 | 13.38928 , 13.83072 |
| _cons       |           |       |       |                      |                   |
| pwless67    | .9785952  | .0619825 | 15.79 | 0.000                | .8571117 , 1.100079 |
| Alien67 _cons |           | 14.67 | .1001814 | 146.43 | 0.000 | 14.47365 , 14.86635 |
| anomia71    | 1 (constrained) | 14.13 | .1159036 | 121.91 | 0.000 | 13.90283 , 14.35717 |
| _cons       |           |       |       |                      |                   |
| pwless71    | .9217508  | .0597225 | 15.43 | 0.000                | .8046968 , 1.038805  |
| Alien71 _cons |           | 14.9  | .1034517 | 144.03 | 0.000 | 14.69724 , 15.10276 |
| educ66      | 1 (constrained) | 10.9 | .1014894 | 107.40 | 0.000 | 10.70108 , 11.09892 |
| _cons       |           |       |       |                      |                   |
| occstat66   | 5.22132   | .425596  | 12.27 | 0.000                | 4.387169 , 6.055471 |
| SES         | 37.49     | .6947112 | 53.96 | 0.000                | 36.12839 , 38.85161 |
| _cons       |           |       |       |                      |                   |
| var(e.anomia67) | 4.728874 | .456299  | 3.914024 | 5.713365  |
| var(e.pwless67) | 2.563413 | .4060733 | 1.879225 | 3.4967    |
| var(e.anomia71) | 4.396081 | .5171156 | 3.490904 | 5.535966  |
| var(e.pwless71) | 3.072085 | .4360333 | 3.236049 | 4.057398  |
| var(e.educ66)  | 2.803674 | .5115854 | 2.022437 | 4.099091  |
| var(e.occstat66) | 264.5311 | 18.22483 | 231.1177 | 302.7751  |
| var(e.Alien67) | 4.842059 | .4622537 | 4.015771 | 5.838364  |
| var(e.Alien71) | 4.084249 | .4038995 | 3.364613 | 4.957802  |
| var(SES)     | 6.796014 | .6524866 | 5.630283 | 8.203105  |
|              |           |       |       |                      |                   |
| cov(e.anomia67, e.anomia71) | 1.622024 | .3154267 | 5.14 | 0.000                | 1.003799 , 2.240249 |
| cov(e.pwless67, e.pwless71) | .3399961 | .2627541 | 1.29 | 0.196                | -.1749925 , .8549847 |

#### LR test of model vs. saturated: chi²(4) = 4.78, Prob > chi² = 0.3111
Notes:

1. We find the covariance between `e.anomia67` and `e.anomia71` to be significant ($Z = 5.14$).
2. We find the covariance between `e.pwless67` and `e.pwless71` to be insignificant at the 5% level ($Z = 1.29$).
3. The model versus saturated $\chi^2$ test indicates that the model is a good fit.

References


Also see

[SEM] Example 32g — Full structural equation model (generalized response)
[SEM] estat mindices — Modification indices
[SEM] test — Wald test of linear hypotheses
Example 10 — MIMIC model

Description

To demonstrate a MIMIC model, we use the following summary statistics data:

```
. use https://www.stata-press.com/data/r16/sem_mimic1
(Multiple indicators and multiple causes)
. ssd describe
Summary statistics data from https://www.stata-press.com/data/r16/sem_mimic1.dta
obs: 432 Multiple indicators and multip..
vars: 5 31 Oct 2018 08:29
(_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>occpres</td>
<td>occupational prestige, two-digit Dunca..</td>
</tr>
<tr>
<td>income</td>
<td>total family income in units of $2000,..</td>
</tr>
<tr>
<td>s_occpres</td>
<td>subjective occupational prestige</td>
</tr>
<tr>
<td>s_income</td>
<td>subjective income</td>
</tr>
<tr>
<td>s_socstat</td>
<td>subjective overall social status</td>
</tr>
</tbody>
</table>
```

. notes
_dta:
2. Data are also analyzed in Bollen, K. A. 1989, _Structural Equations with Latent Variables_, New York: John Wiley & Sons, Inc.
3. The summary statistics represent 432 white adults included in the sample for the 1969 Gary Area Project for the Institute of Social Research at Indiana University.
4. The three subjective variables are measures of socioeconomic status based on an individuals perception of their own income, occupational prestige, and social status.
5. The income and occpres variables are objective measures of income and occupational prestige, respectively.

See _Structural models 10: MIMIC models_ in [SEM] _Intro 5_ for background.

Remarks and examples

Remarks are presented under the following headings:

- **Fitting the MIMIC model**
- **Fitting the MIMIC model with the Builder**
- **Evaluating the residuals with estat residuals**
- **Performing likelihood-ratio tests with lrtest**
Fitting the MIMIC model

Based on the data referenced above, Bollen (1989, 397–399) fits a MIMIC model, the path diagram of which is

Bollen includes paths that he constrains and we do not show. Our model is nonetheless equivalent to the one he shows. In his textbook, Bollen illustrates various ways the same model can be written.
. sem (SubjSES -> s_income s_occpres s_socstat) (SubjSES <- income occpres)

Endogenous variables
Measurement: s_income s_occpres s_socstat
Latent: SubjSES
Exogenous variables
Observed: income occpres

Fitting target model:
Iteration 0:  log likelihood = -4252.1834 (not concave)
Iteration 1:  log likelihood = -4022.8854 (not concave)
Iteration 2:  log likelihood = -3994.3867
Iteration 3:  log likelihood = -3974.6636
Iteration 4:  log likelihood = -3978.7751 (not concave)
Iteration 5:  log likelihood = -3974.6636
Iteration 6:  log likelihood = -3972.0269
Iteration 7:  log likelihood = -3971.9238

Structural equation model  Number of obs  =  432
Estimation method  =  ml
Log likelihood  = -3971.9236

( 1) [s_income]SubjSES = 1

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.  Std. Err.   z  P&gt;</td>
</tr>
<tr>
<td>Structural</td>
<td></td>
</tr>
<tr>
<td>SubjSES</td>
<td></td>
</tr>
<tr>
<td>income</td>
<td>.082732   .0138498  5.97  0.000  .0555869  .1098772</td>
</tr>
<tr>
<td>occpres</td>
<td>.0046275  .0012464  3.71  0.000  .0021847  .0070704</td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>s_income</td>
<td></td>
</tr>
<tr>
<td>SubjSES</td>
<td>1 (constrained)</td>
</tr>
<tr>
<td>_cons</td>
<td>.9612091  .0794151  12.10  0.000  .8055583  1.116886</td>
</tr>
<tr>
<td>s_occpres</td>
<td></td>
</tr>
<tr>
<td>SubjSES</td>
<td>.7301352  .0832915  8.77  0.000  .5668869  .8933835</td>
</tr>
<tr>
<td>_cons</td>
<td>1.114563  .0656195  16.99  0.000  .9859512  1.243175</td>
</tr>
<tr>
<td>s_socstat</td>
<td></td>
</tr>
<tr>
<td>SubjSES</td>
<td>.9405161  .0934855  10.06  0.000  .7572878  1.123744</td>
</tr>
<tr>
<td>_cons</td>
<td>1.002113  .0706576  14.18  0.000  .863627  1.1406</td>
</tr>
<tr>
<td>var(e.s_income)</td>
<td>.2087546  .0254098     .1644474  .2649996</td>
</tr>
<tr>
<td>var(e.s_occpres)</td>
<td>.2811852  .0228914     .2397153  .3298291</td>
</tr>
<tr>
<td>var(e.s_socstat)</td>
<td>.1807129  .0218405     .1425987  .2290146</td>
</tr>
<tr>
<td>var(e.SubjSES)</td>
<td>.1860097  .0270476     .1398822  .2473481</td>
</tr>
</tbody>
</table>

LR test of model vs. saturated: chi2(4)  =  26.65, Prob > chi2  =  0.0000

Notes:

1. In this model, there are three observed variables that record the respondents’ ideas of their perceived socioeconomic status (SES). One is the respondent’s general idea of his or her SES (s_socstat); another is based on the respondent’s income (s_income); and the last is based on the respondent’s occupational prestige (s_occpres). Those three variables form the latent variable SubjSES.

2. The other two observed variables are the respondent’s income (income) and occupation, the latter measured by the two-digit Duncan SEI scores for occupations (occpres). These two variables are treated as predictors of SubjSES.
3. In the model, item 1 is viewed as subjective and item 2 is viewed as objective.

4. All variables are statistically significant at the 5% level, but the model versus saturated test suggests that we are not modeling the covariances well.

**Fitting the MIMIC model with the Builder**

Use the diagram above for reference.

1. Open the dataset.
   In the Command window, type
   ```stata
   . use https://www.stata-press.com/data/r16/sem_mimic1
   ```

2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Create the measurement component for subject socioeconomic status.
   Select the Add measurement component tool, and then click near the center of the diagram.
   In the resulting dialog box,
   a. change the *Latent variable name* to **SubjSES**;
   b. select `s_income`, `s_occpress`, and `s_socstat` by using the *Measurement variables* control;
   c. select `Right` in the *Measurement direction* control;
   d. click on **OK**.
   If you wish, move this component by clicking on any variable and dragging it.

4. Create the two variables for the formative indicators of socioeconomic status.
   a. Select the Add observed variable tool, and then click in the diagram to add the new variable slightly above and to the left of **SubjSES**. After adding it, you can click inside the rectangle and move the variable if you wish.
   b. In the Contextual Toolbar, select `occpress` by using the *Variable* control.
   c. Continuing with the tool, click in the diagram to add another new variable below the `occpress` variable.
   d. In the Contextual Toolbar, select `income` by using the *Variable* control.

5. Create the paths for the formative indicators of socioeconomic status.
   a. Select the Add path tool, .
   b. Click in the lower-right quadrant of the `occpress` rectangle (it will highlight when you hover over it), and drag a path to the upper-left quadrant of the **SubjSES** oval (it will highlight when you can release to connect the path).
   c. Continuing with the tool, click in the upper-right quadrant of the `income` rectangle, and drag a path to the lower-left quadrant of the **SubjSES** oval.
6. Clean up the direction of the errors.

The error on \texttt{SubjSES} is likely to have been created atop one of the existing paths. Choose the Select tool, \(\texttt{[Select]}\), and then click in the \texttt{SubjSES} oval. Click on one of the \textbf{Error rotation} buttons, \(\texttt{[Error rotation]}\), in the Contextual Toolbar until the error is where you want it.

7. Estimate.

Click on the \textbf{Estimate} button, \(\texttt{[Estimate]}\), in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{SEM estimation options} dialog box.

Tip: See the tips of \textbf{Example 9} to make creating paths somewhat easier than described above.

You can open a completed diagram in the Builder by typing

\[. \texttt{webgetsem sem\_mimic1}\]

\textbf{Evaluating the residuals with estat residuals}

Remember that \textit{SEM} fits covariances and means. Residuals in the \textit{SEM} sense thus refer to covariances and means. If we are not fitting well, we can examine the residuals.

\[. \texttt{estat residuals, normalized}\]

\textbf{Residuals of observed variables}

\begin{tabular}{l|cccc}
\multicolumn{1}{c|}{Mean residuals} & \textit{s\_income} & \textit{s\_occpres} & \textit{s\_socstat} & \textit{income} & \textit{occpres} \\
\hline
raw & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\
normalized & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\
\end{tabular}

\textbf{Covariance residuals}

\begin{tabular}{l|cccc}
\textit{s\_income} & \textit{s\_occpres} & \textit{s\_socstat} & \textit{income} & \textit{occpres} \\
\hline
\textit{s\_income} & 0.000 & & & \\
\textit{s\_occpres} & -0.009 & 0.000 & & \\
\textit{s\_socstat} & 0.000 & 0.008 & 0.000 & \\
\textit{income} & 0.101 & -0.079 & -0.053 & 0.000 \\
\textit{occpres} & -0.856 & 1.482 & 0.049 & 0.000 & 0.000 \\
\end{tabular}

\textbf{Normalized covariance residuals}

\begin{tabular}{l|cccc}
\textit{s\_income} & \textit{s\_occpres} & \textit{s\_socstat} & \textit{income} & \textit{occpres} \\
\hline
\textit{s\_income} & 0.000 & & & \\
\textit{s\_occpres} & -0.425 & 0.000 & & \\
\textit{s\_socstat} & 0.008 & 0.401 & 0.000 & \\
\textit{income} & 1.362 & -1.137 & -0.771 & 0.000 \\
\textit{occpres} & -1.221 & 2.234 & 0.074 & 0.000 & 0.000 \\
\end{tabular}

\textbf{Notes:}

1. The residuals can be partitioned into two subsets: mean residuals and covariance residuals.
2. The \texttt{normalized} option caused the normalized residuals to be displayed.
3. Concerning mean residuals, the raw residuals and the normalized residuals are shown on a separate line of the first table.
4. Concerning covariance residuals, the raw residuals and the normalized residuals are shown in separate tables.

5. Distinguish between normalized residuals and standardized residuals. Both are available from `estat residuals`; if we wanted standardized residuals, we would have specified the `standardized` option instead of or along with `normalized`.

6. Both normalized and standardized residuals attempt to adjust the residuals in the same way. The normalized residuals are always valid, but they do not follow a standard normal distribution. The standardized residuals do follow a standard normal distribution if they can be calculated; otherwise, they will equal missing values. When both can be calculated (equivalent to both being appropriate), the normalized residuals will be a little smaller than the standardized residuals.

7. The normalized covariance residuals between `income` and `s_income` and between `occpres` and `s_occpres` are large.

**Performing likelihood-ratio tests with `lrtest`**

Thus Bollen (1989, 397–399) suggests adding a direct path from the objective measures to the corresponding subjective measures. We are about to fit the model

```
(SubjSES -> s_income s_occpres s_socstat) ///
(SubjSES <- income occpres) ///
(s_income <- income) /// <- new
(s_occpres <- occpres) /// <- new
```

For no other reason than we want to demonstrate the likelihood-ratio test, we will then use `lrtest` rather than `test` to test the joint significance of the new paths. `lrtest` compares the likelihood values of two fitted models. Thus we will use `lrtest` to compare this new model with the one above. To do that, we must plan ahead and store in memory the currently fit model:

```
. estimates store mimic1
```

Alternatively, we could skip that and calculate the joint significance of the two new paths by using a Wald test and the `test` command.

In any case, having stored the current estimates under the name `mimic1`, we can now fit our new model:

```
. sem (SubjSES -> s_income s_occpres s_socstat)
> (SubjSES <- income occpres)
> (s_income <- income)
> (s_occpres <- occpres)
```

**Endogenous variables**

Observed: `s_income` `s_occpres`

Measurement: `s_socstat`

**Latent:** `SubjSES`

**Exogenous variables**

Observed: `income` `occpres`

**Fitting target model:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-4267.0974</td>
</tr>
<tr>
<td>1</td>
<td>-4022.6745</td>
</tr>
<tr>
<td>2</td>
<td>-3977.0648</td>
</tr>
<tr>
<td>3</td>
<td>-3962.9229</td>
</tr>
<tr>
<td>4</td>
<td>-3962.1604</td>
</tr>
<tr>
<td>5</td>
<td>-3960.8404</td>
</tr>
<tr>
<td>6</td>
<td>-3960.7133</td>
</tr>
<tr>
<td>7</td>
<td>-3960.7111</td>
</tr>
<tr>
<td>8</td>
<td>-3960.7111</td>
</tr>
</tbody>
</table>
Example 10 — MIMIC model

Structural equation model

Number of obs = 432
Estimation method = ml
Log likelihood = -3960.7111

( 1) [s_income]SubjSES = 1

| OIM                  | Coef.     | Std. Err. | z   | P>|z|   | [95% Conf. Interval] |
|---------------------|-----------|-----------|-----|--------|---------------------|
| Structural          |           |           |     |        |                     |
| s_income            |           |           |     |        |                     |
| SubjSES             | 1 (constrained) |       |     |        |                     |
| income              | 0.0532425 | 0.0142862 | 3.73| 0.000  | 0.025242 .081243    |
| _cons               | 0.8825314 | 0.0781685 | 11.29| 0.000  | 0.7293239 1.035739  |
| s_occpres           |           |           |     |        |                     |
| SubjSES             |           |           |     |        |                     |
| occpres             | 0.783781 | 0.1011457 | 7.75| 0.000  | 0.585539 .982023    |
| _cons               | 1.06586  | 0.0696058 | 15.31| 0.000  | 0.9294353 1.202285  |
| Measurement         |           |           |     |        |                     |
| s_socstat           |           |           |     |        |                     |
| SubjSES             |           |           |     |        |                     |
| income              | 0.0538025 | 0.0129158 | 4.17| 0.000  | 0.028488 .0791171   |
| occpres             | 0.0045201 | 0.0013552 | 3.34| 0.001  | 0.0018641 .0071762  |
| _cons               | 1.06586  | 0.0696058 | 15.31| 0.000  | 0.9294353 1.202285  |
| var(e.s_in~e)       | .2292697  | .0248905  |     |        | .1853261 .2836329   |
| var(e.s_oc~s)       | .2773786  | .0223972  |     |        | .2367783 .3249407   |
| var(e.s_so~t)       | .1459009  | .0282228  |     |        | .0998556 .2131785   |
| var(e.Subj~S)       | .1480275  | .0278381  |     |        | .1023918 .2140029   |

LR test of model vs. saturated: chi2(2) = 4.22, Prob > chi2 = 0.1211

Now we can perform the likelihood-ratio test:

```
  . lrtest mimic1 .
  Likelihood-ratio test LR chi2(2) = 22.42 Prob > chi2 = 0.0000
```

Notes:

1. The syntax of `lrtest` is `lrtest modelname1 modelname2`. We specified the first model name as `mimic1`, the model we previously stored. We specified the second model name as `period (.)`, meaning the model most recently fit. The order in which we specify the names is irrelevant.

2. We find the two added paths to be extremely significant.

Reference


Also see

- [SEM] Example 36g — MIMIC model (generalized response)
- [SEM] sem — Structural equation model estimation command
- [SEM] estat residuals — Display mean and covariance residuals
- [SEM] lrtest — Likelihood-ratio test of linear hypothesis
## Example 11 — estat framework

### Description

To demonstrate `estat framework`, which displays results in Bentler–Weeks form, we continue where [SEM] Example 10 left off:

```
use https://www.stata-press.com/data/r16/sem_mimic1
.ssd describe
.notes
.sem (SubjSES -> s_income s_occpres s_socstat) ///
    (SubjSES <- income occpres)
.estat residuals, normalized
.estimates store mimic1
.sem (SubjSES -> s_income s_occpres s_socstat) ///
    (SubjSES <- income occpres) ///
    (s_income <- income) ///
    (s_occpres <- occpres)
lrtest mimic1
```

See *Structural models 10: MIMIC models* in [SEM] Intro 5 for background.

### Remarks and examples

If you prefer to see SEM results reported in Bentler–Weeks form, type `estat framework` after estimating with `sem`. Many people find Bentler–Weeks form helpful in understanding how the model is fit.

[SEM] Example 10 ended by fitting

```
.sem (SubjSES -> s_income s_occpres s_socstat) ///
    (SubjSES <- income occpres) ///
    (s_income <- income) ///
    (s_occpres <- occpres)
```

In Bentler–Weeks form, the output appears as

```
estat framework, fitted

Endogenous variables on endogenous variables

<table>
<thead>
<tr>
<th>Beta</th>
<th>observed</th>
<th>latent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s_income</td>
<td>s_occpres</td>
</tr>
<tr>
<td>observed</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_income</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_occpres</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_socstat</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

236
### Exogenous variables on endogenous variables

<table>
<thead>
<tr>
<th>Gamma</th>
<th>observed</th>
<th>income</th>
<th>occpres</th>
</tr>
</thead>
<tbody>
<tr>
<td>observed</td>
<td>s_income</td>
<td>0.0532425</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>s_occpres</td>
<td>0</td>
<td>0.0045201</td>
</tr>
<tr>
<td></td>
<td>s_socstat</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>latent</td>
<td>SubjSES</td>
<td>0.0538025</td>
<td>0.0034324</td>
</tr>
</tbody>
</table>

### Covariances of error variables

<table>
<thead>
<tr>
<th>Psi</th>
<th>observed</th>
<th>e.s_inc~e</th>
<th>e.s_occ~s</th>
<th>e.s_soc~t</th>
<th>latent</th>
<th>e.SubjSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>observed</td>
<td>e.s_income</td>
<td>0.2292697</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>e.s_occpres</td>
<td>0</td>
<td>0.2773786</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>e.s_socstat</td>
<td>0</td>
<td>0</td>
<td>0.1459009</td>
<td></td>
<td></td>
</tr>
<tr>
<td>latent</td>
<td>e.SubjSES</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1480275</td>
<td></td>
</tr>
</tbody>
</table>

### Intercepts of endogenous variables

<table>
<thead>
<tr>
<th>alpha</th>
<th>observed</th>
<th>s_income</th>
<th>s_occpres</th>
<th>s_socstat</th>
<th>latent</th>
<th>SubjSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>_cons</td>
<td></td>
<td>0.8825314</td>
<td>1.06586</td>
<td>1.07922</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### Covariances of exogenous variables

<table>
<thead>
<tr>
<th>Phi</th>
<th>observed</th>
<th>income</th>
<th>occpres</th>
</tr>
</thead>
<tbody>
<tr>
<td>observed</td>
<td>income</td>
<td>4.820021</td>
<td></td>
</tr>
<tr>
<td></td>
<td>occpres</td>
<td>13.62431</td>
<td>451.6628</td>
</tr>
</tbody>
</table>

### Means of exogenous variables

<table>
<thead>
<tr>
<th>kappa</th>
<th>observed</th>
<th>income</th>
<th>occpres</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td></td>
<td>5.04</td>
<td>36.698</td>
</tr>
</tbody>
</table>
Fitted covariances of observed and latent variables

<table>
<thead>
<tr>
<th>Sigma</th>
<th>observed</th>
<th>latent</th>
<th>observed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s_income</td>
<td>SubjSES</td>
<td>income</td>
</tr>
<tr>
<td>observed</td>
<td>.4478609</td>
<td>.3060932</td>
<td>4.820021</td>
</tr>
<tr>
<td>s_occpres</td>
<td>.1614446</td>
<td>.3014937</td>
<td>5.04</td>
</tr>
<tr>
<td>s_socstat</td>
<td>.225515</td>
<td>.3659463</td>
<td>1.54</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sigma</th>
<th>observed</th>
<th>latent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>income</td>
<td>.3971264</td>
</tr>
<tr>
<td>occpres</td>
<td>3.008694</td>
<td>3.831184</td>
</tr>
<tr>
<td></td>
<td>2.729776</td>
<td>2.283302</td>
</tr>
<tr>
<td></td>
<td>13.62431</td>
<td>13.62431</td>
</tr>
</tbody>
</table>

Fitted means of observed and latent variables

<table>
<thead>
<tr>
<th>mu</th>
<th>observed</th>
<th>latent</th>
<th>observed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s_income</td>
<td>SubjSES</td>
<td>income</td>
</tr>
<tr>
<td>mu</td>
<td>1.548</td>
<td>.3971264</td>
<td>5.04</td>
</tr>
<tr>
<td>mu</td>
<td>1.543</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mu</td>
<td>1.554</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:

1. Bentler–Weeks form is a vector and matrix notation for the estimated parameters of the model. The matrices are known as $\beta$, $\Gamma$, $\Psi$, $\alpha$, $\Phi$, and $\kappa$. Those Greek names are spelled out in the labels, along with a header stating what each contains.

2. We specified estat framework option fitted. That caused estat framework to list one more matrix and one more vector at the end: $\Sigma$ and $\mu$. These two results are especially interesting to those wishing to see the ingredients of the residuals reported by estat residuals.

3. One of the more useful results reported by estat framework, fitted is the $\Sigma$ matrix, which reports all estimated covariances in a readable format and includes the model-implied covariances that do not appear in sem’s ordinary output.

4. estat framework also allows the standardized option if you want standardized output.

Also see

[SEM] Example 10 — MIMIC model
[SEM] estat framework — Display estimation results in modeling framework
Example 12 — Seemingly unrelated regression

Description

sem can be used to estimate seemingly unrelated regression. We will use auto.dta, which surely needs no introduction:

```
. sysuse auto
(1978 Automobile Data)
```

See Structural models 11: Seemingly unrelated regression (SUR) in [SEM] Intro 5.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the seemingly unrelated regression model
- Fitting the model with the Builder

Fitting the seemingly unrelated regression model

We fit the following model:

```
. sem (price <- foreign mpg displacement) > (weight <- foreign length),
    > cov(e.price*e.weight)
```

Endogenous variables
Observed: price weight
Exogenous variables
Observed: foreign mpg displacement length

Fitting target model:
Iteration 0:  log likelihood =  -2150.9983
Iteration 1:  log likelihood =  -2138.5739
Iteration 2:  log likelihood =  -2133.3461
Iteration 3:  log likelihood =  -2133.1979
Iteration 4:  log likelihood =  -2133.1956
Iteration 5:  log likelihood =  -2133.1956
Example 12 — Seemingly unrelated regression

Structural equation model

Number of obs = 74
Estimation method = ml
Log likelihood = -2133.1956

| OIM                      | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------------------------|-------|-----------|-------|------|---------------------|
| Structural               |       |           |       |      |                     |
| price                    |       |           |       |      |                     |
| foreign                  | 2940.929 | 724.7311 | 4.06  | 0.000| 1520.482 4361.376   |
| mpg                      | -105.0163 | 57.93461 | -1.81 | 0.070| -218.566 8.53347    |
| displace=t              | 17.22083  | 4.5941    | 3.75  | 0.000| 8.216558 26.2251    |
| _cons                    | 4129.866  | 1984.253  | 2.08  | 0.037| 240.8022 8018.931   |
| weight                   |       |           |       |      |                     |
| foreign                  | -153.2515 | 76.21732 | -2.01 | 0.044| -302.6347 -3.868275 |
| length                   | 30.73507  | 1.584743  | 19.39 | 0.000| 27.62903 33.84111  |
| _cons                    | -2711.096 | 312.6813  | -8.67 | 0.000| -3323.94 -2098.252 |
| var(e.price)             | 4732491  | 801783.1  |       |      | 3395302 6596312    |
| var(e.weight)            | 60253.09  | 9933.316  |       |      | 43616.45 83235.44  |
| cov(e.price, e.weight)   | 209268   | 73909.54  | 2.83  | 0.005| 64407.92 354128    |

LR test of model vs. saturated: chid2(3) = 38.86, Prob > chi2 = 0.0000

Notes:

1. Point estimates are the same as reported by

   . sureg (price foreign mpg displ) (weight foreign length), isure

   sureg’s isure option is required to make sureg iterate to the maximum likelihood estimate.

2. If you wish to compare the estimated variances and covariances after estimation by sureg, type

   . matrix list e(Sigma)

   sureg does not estimate standard errors on variances and covariances.

3. Standard errors will be different between sem and sureg. In this case, there is no reason to prefer one set of standard errors over the other, and standard errors are asymptotically equivalent. This is a case of exogenous variables only on the right-hand side. When the model being fit is recursive, standard errors produced by sem are better than those from sureg, both asymptotically and in finite samples.

4. One reason you might want to use sem is that sem will provide robust standard errors whereas sureg does not.

5. Multivariate regression can be viewed as seemingly unrelated regression. You just need to specify the same regressors for each equation. In that case, standard errors reported by sem will be the same as those reported by mvreg if one applies the multiplicative $\sqrt{(N - p - 1)/N}$ degree-of-freedom adjustment.
Fitting the model with the Builder

Use the diagram above for reference.

1. Open the dataset.
   In the Command window, type
   . use https://www.stata-press.com/data/r16/auto

2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Change the size of the observed variables’ rectangles.
   From the SEM Builder menu, select Settings > Variables > All observed....
   In the resulting dialog box, change the first size to .85 and click on OK.

4. Create the four independent variables.
   Select the Add observed variables set tool, ☐, and then click in the diagram about one-fourth of the way in from the left and one-fourth of the way up from the bottom.
   In the resulting dialog box,
   a. select the Select variables radio button (it may already be selected);
   b. use the Variables control to select the four variables in this order: mpg, displacement, foreign, and length;
   c. select Vertical in the Orientation control;
   d. click on OK.
   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the two dependent variables.
   Select the Add observed variables set tool, ☐, and then click about two-thirds of the way in from the left and vertically aligned with the top of the length rectangle.
   In the resulting dialog box,
   a. select the Select variables radio button (it may already be selected);
   b. use the Variables control to select the variables price and weight;
   c. select Vertical in the Orientation control;
   d. select the Distances tab;
   e. select .5 (inch) from the Distance between variables control;
   f. click on OK.
   If you wish, move the set of variables by clicking on any variable and dragging it.

6. Create paths from the independent variables to the dependent variables.
   a. Select the Add path tool, −.
   b. Click in the right side of the mpg rectangle (it will highlight when you hover over it), and drag a path to the left side of the price rectangle (it will highlight when you can release to connect the path).
c. Continuing with the tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable:

- displacement -> price
- foreign -> price
- foreign -> weight
- length -> weight

7. Correlate the error terms.
   a. Select the Add covariance tool, $\sim$.
   b. Click in the $\epsilon_1$ circle (it will highlight when you hover over it), and drag a covariance to the $\epsilon_2$ circle (it will highlight when you can release to connect the covariance).

8. Clean up.

   If you do not like where a path has been connected to its variables, use the Select tool, $\select$, to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint. Similarly, you can change where the covariance connects to the error terms by clicking on the covariance and dragging the endpoint. You can also change the bow of the covariance by clicking on the covariance and dragging the control point that extends from one end of the selected covariance.


   Click on the Estimate button, $\estimate$, in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

   You can open a completed diagram in the Builder by typing

   . webgetsem sem_sureg

Also see

[SEM] Example 13 — Equation-level Wald test

[SEM] sem — Structural equation model estimation command
Description

We demonstrate estat eqtest. See [SEM] Intro 7 and [SEM] estat eqtest.

This example picks up where [SEM] Example 12 left off:

```
. use https://www.stata-press.com/data/r16/auto
. sem (price <- foreign mpg displacement) ///
    (weight <- foreign length), ///
    cov(e.price*e.weight)
```

Remarks and examples

We have fit a two-equation model with equations for endogenous variables `price` and `weight`. There happen to be two equations, the model happens to be a seemingly unrelated regression, and the endogenous variables happen to be observed, but none of that is important right now.

`estat eqtest` displays equation-by-equation Wald tests that all coefficients excluding the intercepts are 0.

```
. estat eqtest

Wald tests for equations

<table>
<thead>
<tr>
<th></th>
<th>chi2</th>
<th>df</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>observed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>price</td>
<td>36.43</td>
<td>3</td>
<td>0.0000</td>
</tr>
<tr>
<td>weight</td>
<td>633.34</td>
<td>2</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
```

Note:

1. The null hypothesis for this test is that the coefficients other than the intercepts are 0. We can reject that null hypothesis for each equation.

Also see

- [SEM] Example 12 — Seemingly unrelated regression
- [SEM] Intro 7 — Postestimation tests and predictions
- [SEM] estat eqtest — Equation-level tests that all coefficients are zero
Example 14 — Predicted values

Description

We demonstrate the use of predict. See [SEM] Intro 7 and [SEM] predict after sem.

This example picks up where the first part of [SEM] Example 1 left off:

```
.use https://www.stata-press.com/data/r16/sem_1fmm
.sem (x1 x2 x3 x4 <- X)
```

Remarks and examples

predict can create new variables containing predicted values of 1) observed endogenous variables, 2) latent variables, whether endogenous or exogenous, and 3) latent endogenous variables. In the case of latent variables, item 2 corresponds to the factor score and item 3 is the linear prediction.

Below we demonstrate 1 and 2:

```
.predict x1hat x2hat, xb(x1 x2)
predict Xhat, latent(X)
```

You specify options on predict to specify what you want predicted and how. Because of the differing options, the two commands could not have been combined into one command.

Our dataset now contains three new variables. Below we compare the three variables with the original x1 and x2 by using first summarize and then correlate:

```
.summarize x1 x1hat x2 x2hat Xhat
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>500</td>
<td>99.518</td>
<td>14.35402</td>
<td>60</td>
<td>137</td>
</tr>
<tr>
<td>x1hat</td>
<td>500</td>
<td>99.518</td>
<td>9.363112</td>
<td>71.45533</td>
<td>126.7325</td>
</tr>
<tr>
<td>x2</td>
<td>500</td>
<td>99.954</td>
<td>14.1939</td>
<td>52</td>
<td>140</td>
</tr>
<tr>
<td>x2hat</td>
<td>500</td>
<td>99.954</td>
<td>9.674426</td>
<td>70.95827</td>
<td>128.0733</td>
</tr>
<tr>
<td>Xhat</td>
<td>500</td>
<td>1.03e-08</td>
<td>9.363112</td>
<td>-28.06267</td>
<td>27.21449</td>
</tr>
</tbody>
</table>

Notes:

1. Means of x1hat and x1 are identical; means of x2hat and x2 are identical.
2. The standard deviation of x1hat is less than that of x1; the standard deviation of x2hat is less than that of x2. Some of the variation in x1 and x2 is not explained by the model.
3. Standard deviations of x1hat and Xhat are equal. This is because in
   \[ x_1 = b_0 + b_1 X + e_1 \]
   coefficient \( b_1 \) was constrained to be equal to 1 because of the anchoring normalization constraint; see Identification 2: Normalization constraints (anchoring) in [SEM] Intro 4.
The mean of $X_{\text{hat}}$ in the model above is $1.03\times10^{-8}$ rather than 0. Had we typed

```
predict double Xhat, latent(X)
```

the mean would have been $-1.17\times10^{-16}$.

```
correlate x1 x1hat x2 x2hat Xhat
(obs=500)
```

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x1hat</th>
<th>x2</th>
<th>x2hat</th>
<th>Xhat</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x1hat</td>
<td>0.6705</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>0.4537</td>
<td>0.7007</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2hat</td>
<td>0.6705</td>
<td>1.0000</td>
<td>0.7007</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>Xhat</td>
<td>0.6705</td>
<td>1.0000</td>
<td>0.7007</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Notes:

1. Both $x_{1\text{hat}}$ and $x_{2\text{hat}}$ correlate 1 with $X_{\text{hat}}$. That is because both are linear functions of $X_{\text{hat}}$ alone.
2. That $x_{1\text{hat}}$ and $x_{2\text{hat}}$ correlate 1 is implied by item 1, directly above.
3. That $X_{\text{hat}}$, $x_{1\text{hat}}$, and $x_{2\text{hat}}$ all have the same correlation with $x_{1}$ and with $x_{2}$ is also implied by item 1, directly above.

Also see

[SEM] Example 1 — Single-factor measurement model
[SEM] Intro 7 — Postestimation tests and predictions
[SEM] predict after sem — Factor scores, linear predictions, etc.
**Example 15 — Higher-order CFA**

**Description**

`sem` can be used to estimate higher-order confirmatory factor analysis models.

```
use https://www.stata-press.com/data/r16/sem_hcfa1
(Higher-order CFA)
.ssd describe
Summary statistics data from
https://www.stata-press.com/data/r16/sem_hcfa1.dta
obs: 251 Higher-order CFA
vars: 16 25 May 2018 11:26
(_dta has notes)
variable name variable label
phyab1 Physical ability 1
phyab2 Physical ability 2
phyab3 Physical ability 3
phyab4 Physical ability 4
appear1 Appearance 1
appear2 Appearance 2
appear3 Appearance 3
appear4 Appearance 4
peerrel1 Relationship w/ peers 1
peerrel2 Relationship w/ peers 2
peerrel3 Relationship w/ peers 3
peerrel4 Relationship w/ peers 4
parrel1 Relationship w/ parent 1
parrel2 Relationship w/ parent 2
parrel3 Relationship w/ parent 3
parrel4 Relationship w/ parent 4
```

```
.notes
_dta:
1. Summary statistics data from Marsh, H. W. and Hocevar, D., 1985,
"Application of confirmatory factor analysis to the study of
self-concept: First- and higher order factor models and their invariance
across groups", _Psychological Bulletin_, 97: 562-582.
2. Summary statistics based on 251 students from Sydney, Australia in Grade
5.
3. Data collected using the Self-Description Questionnaire and includes
sixteen subscales designed to measure nonacademic traits: four intended
to measure physical ability, four intended to measure physical
appearance, four intended to measure relations with peers, and four
intended to measure relations with parents.
```

See `Higher-order CFA models` in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

Fitting the model
  Fitting the model with the Builder

Fitting the model

We fit the following model:
Example 15 — Higher-Order CFA

```
. sem (Phys -> phyab1 phyab2 phyab3 phyab4)
> (Appear -> appear1 appear2 appear3 appear4)
> (Peer -> peerrel1 peerrel2 peerrel3 peerrel4)
> (Par -> parrel1 parrel2 parrel3 parrel4)
> (Nonacad -> Phys Appear Peer Par)

Endogenous variables
Measurement: phyab1 phyab2 phyab3 phyab4 appear1 appear2 appear3 appear4 peerrel1 peerrel2 peerrel3 peerrel4 parrel1 parrel2 parrel3 parrel4
Latent: Phys Appear Peer Par

Exogenous variables
Latent: Nonacad

Fitting target model:
Iteration 0:  log likelihood = -7686.6699 (not concave)
Iteration 1:  log likelihood = -7643.7387 (not concave)
Iteration 2:  log likelihood = -7616.2966 (not concave)
Iteration 3:  log likelihood = -7597.6133
Iteration 4:  log likelihood = -7588.9515
Iteration 5:  log likelihood = -7585.3162
Iteration 6:  log likelihood = -7584.8125
Iteration 7:  log likelihood = -7584.7885
Iteration 8:  log likelihood = -7584.7881

Structural equation model
Number of obs = 251
Estimation method = ml
Log likelihood = -7584.7881

( 1) [phyab1]Phys = 1
( 2) [appear1]Appear = 1
( 3) [peerrel1]Peer = 1
( 4) [parrel1]Par = 1
( 5) [Phys]Nonacad = 1

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>Structural</td>
<td></td>
</tr>
<tr>
<td>Phys</td>
<td></td>
</tr>
<tr>
<td>Nonacad</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Appear</td>
<td></td>
</tr>
<tr>
<td>Nonacad</td>
<td>2.202491</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer</td>
<td></td>
</tr>
<tr>
<td>Nonacad</td>
<td>1.448035</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Par</td>
<td></td>
</tr>
<tr>
<td>Nonacad</td>
<td>.569956</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>phyab1</td>
<td></td>
</tr>
<tr>
<td>Phys</td>
<td>1</td>
</tr>
<tr>
<td>_cons</td>
<td>8.2</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>phyab2</td>
<td></td>
</tr>
<tr>
<td>Phys</td>
<td>.9332477</td>
</tr>
<tr>
<td>_cons</td>
<td>8.23</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Variable</th>
<th>Parameter 1</th>
<th>Parameter 2</th>
<th>Parameter 3</th>
<th>Parameter 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>phyab3</td>
<td>1.529936</td>
<td>.1573845</td>
<td>9.72</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.17</td>
<td>.1303953</td>
<td>62.66</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.91443</td>
<td>8.42557</td>
<td></td>
<td></td>
</tr>
<tr>
<td>phyab4</td>
<td>1.325641</td>
<td>.1338053</td>
<td>9.91</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.56</td>
<td>.1146471</td>
<td>74.66</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.335296</td>
<td>8.784704</td>
<td></td>
<td></td>
</tr>
<tr>
<td>appear1</td>
<td></td>
<td></td>
<td>1 (constrained)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7.41</td>
<td>.1474041</td>
<td>50.27</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.121093</td>
<td>7.698907</td>
<td></td>
<td></td>
</tr>
<tr>
<td>appear2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0719</td>
<td>.0821893</td>
<td>13.04</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>.1644123</td>
<td>42.58</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>6.677758</td>
<td>7.322242</td>
<td></td>
<td></td>
</tr>
<tr>
<td>appear3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.035198</td>
<td>.0893075</td>
<td>11.59</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.17</td>
<td>.1562231</td>
<td>45.90</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>6.863808</td>
<td>7.476192</td>
<td></td>
<td></td>
</tr>
<tr>
<td>appear4</td>
<td></td>
<td></td>
<td>.9424492</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>7.4</td>
<td>.1474041</td>
<td>50.20</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.111093</td>
<td>7.688907</td>
<td></td>
<td></td>
</tr>
<tr>
<td>peerrel1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8.81</td>
<td>.1077186</td>
<td>81.79</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.598875</td>
<td>9.021125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>peerrel2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.214379</td>
<td>.1556051</td>
<td>7.80</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.94</td>
<td>.1215769</td>
<td>65.31</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.701714</td>
<td>8.178286</td>
<td></td>
<td></td>
</tr>
<tr>
<td>peerrel3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.667829</td>
<td>.190761</td>
<td>8.74</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.52</td>
<td>.1373248</td>
<td>54.76</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>7.250848</td>
<td>7.789152</td>
<td></td>
<td></td>
</tr>
<tr>
<td>peerrel4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.363627</td>
<td>.159982</td>
<td>8.52</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.29</td>
<td>.1222066</td>
<td>67.84</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.050479</td>
<td>8.529521</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parrel1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9.35</td>
<td>.0825215</td>
<td>113.30</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>9.188261</td>
<td>9.511739</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parrel2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.159754</td>
<td>.184581</td>
<td>6.28</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>9.13</td>
<td>.0988998</td>
<td>92.32</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.93616</td>
<td>9.32384</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parrel3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.035143</td>
<td>.2623826</td>
<td>7.76</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.67</td>
<td>.1114983</td>
<td>77.76</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.451467</td>
<td>8.888533</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parrel4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.651802</td>
<td>.2116151</td>
<td>7.81</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>.0926003</td>
<td>97.19</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>8.818507</td>
<td>9.181493</td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.phyab1)</td>
<td>2.07466</td>
<td>.2075636</td>
<td>1.705244</td>
<td>2.524103</td>
</tr>
<tr>
<td>var(e.phyab2)</td>
<td>2.618638</td>
<td>.252693</td>
<td>2.167386</td>
<td>3.163841</td>
</tr>
<tr>
<td>var(e.phyab3)</td>
<td>1.231013</td>
<td>.2062531</td>
<td>8.864333</td>
<td>1.70954</td>
</tr>
<tr>
<td>var(e.phyab4)</td>
<td>1.019261</td>
<td>.1600644</td>
<td>.7492262</td>
<td>1.386621</td>
</tr>
<tr>
<td>var(e.appe~1)</td>
<td>1.986955</td>
<td>.2711164</td>
<td>1.520699</td>
<td>2.596169</td>
</tr>
</tbody>
</table>
Example 15 — Higher-order CFA

```
var(e.appe~2)  2.801673 .3526427  2.189162 3.585561
var(e.appe~3)  2.41072 .3002627  1.889595 2.277933
var(e.appe~4)  2.374508 .2872554  1.873267 2.390749
var(e.peer~1)  1.866632 .18965  1.529595 2.277933
var(e.peer~2)  2.167766 .2288099  1.762654 2.665984
var(e.peer~3)  1.824346 .2516762  1.392131 2.390749
var(e.peer~4)  1.803918 .2872554  1.431856 2.272659
var(e.parr~1)  1.214141 .18965  1.529595 2.277933
var(e.parr~2)  1.789125 .1748043  1.477322 2.166738
var(e.parr~3)  1.069717 .1767086  1.767086 2.166738
var(e.parr~4)  0.803735 .121231  1.21231  1.077963
var(e.Phys)    .911538 .1933432  0.6014913 1.381403
var(e.Appear)  1.59518 .3704939  1.011838 2.514828
var(e.Peer)    .2368108 .1193956  0.3697854 .600597
var(e.Par)     .3697854 .0915049  0.2276755 .600597
var(Nonacad)   .3858166 .1237638  0.2057449 .7234903
```

LR test of model vs. saturated: chi2(100) = 219.48, Prob > chi2 = 0.0000

Notes:

1. The idea behind this model is that physical ability, appearance, and relationships with peers and parents may be determined by a latent variable containing nonacademic traits. This model was suggested by Bollen (1989, 315).
2. `sem` automatically provided normalization constraints for the first-order factors Phys, Appear, Peer, and Par. Their path coefficients were set to 1.
3. `sem` automatically provided a normalization constraint for the second-order factor Nonacad. Its path coefficient was set to 1.

**Fitting the model with the Builder**

Use the diagram above for reference.

1. Open the dataset.

   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/sem_hcfa1
   ```

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Enlarge the size of the canvas to accommodate the length of the diagram.

   Click on the **Adjust canvas size** button, ⚙️, in the Standard Toolbar, change the second size to 7 (inches), and then click on **OK**.

4. Change the size of the observed variables’ rectangles.

   a. In the **SEM Builder menu**, select **Settings > Variables > All observed....**
   
   b. In the resulting dialog box, change the second size to .25 and click on **OK**.

5. Create the measurement component for physical ability.

   Select the Add measurement component tool, 🌐. Then using the darker one-inch grid lines in the background as a guide, click in the diagram about two inches in from the left and one inch down from the top.
In the resulting dialog box,

a. change the \textit{Latent variable name} to \texttt{Phys};

b. select \texttt{phyab1}, \texttt{phyab2}, \texttt{phyab3}, and \texttt{phyab4} by using the \textit{Measurement variables} control;

c. select \texttt{Left} in the \textit{Measurement direction} control;

d. click on \texttt{OK}.

If you wish, move the component by clicking on any variable and dragging it.

6. Create the remaining first-order measurement components.

a. Repeat the process from item 5, but place the measurement component on the grid line two inches in from the left and about two and one-half inches down from the top. Label the latent variable \texttt{Appear}, and select measurement variables \texttt{appear1}, \texttt{appear2}, \texttt{appear3}, and \texttt{appear4}.

b. Repeat the process from item 5, but place the measurement component on the grid line two inches in from the left and about four inches down from the top. Label the latent variable \texttt{Peer}, and select measurement variables \texttt{peerrel1}, \texttt{peerrel2}, \texttt{peerrel3}, and \texttt{peerrel4}.

c. Repeat the process from item 5, but place the measurement component on the grid line two inches in from the left and about five and one-half inches down from the top. Label the latent variable \texttt{Par}, and select measurement variables \texttt{parrel1}, \texttt{parrel2}, \texttt{parrel3}, and \texttt{parrel4}.

7. Create the second-order latent variable.

a. Select the Add latent variable tool, \texttt{○}, and then click in the diagram about two inches in from the right and vertically centered between the \texttt{Appear} and \texttt{Peer} latent variables.

b. In the Contextual Toolbar, type \texttt{Nonacad} in the \textit{Name} control and press \texttt{Enter}.

8. Create paths from \texttt{Nonacad} to each of the first-order latent variables.

a. Click in the upper-left quadrant of the \texttt{Nonacad} oval (it will highlight when you hover over it), and drag a path to the lower-left quadrant of the \texttt{Phys} oval (it will highlight when you can release to connect the path).

b. Continuing with the \texttt{−} tool, create the following paths by clicking first on the left side of the \texttt{Nonacad} variable and dragging to the right side of the first-order latent variable.

\begin{verbatim}
Nonacad -> Appear
Nonacad -> Peer
Nonacad -> Par
\end{verbatim}

9. Clean up the direction of the errors.

We want the errors for each of the latent variables to be below the latent variable. The errors for \texttt{Phys}, \texttt{Appear}, and \texttt{Peer} are likely to have been created in other directions.

a. Choose the Select tool, \texttt{\textbullet}.

b. Click in the \texttt{Phys} oval.

c. Click on one of the \textit{Error rotation} buttons, \texttt{↔}, in the Contextual Toolbar until the error is below the latent variable.

Repeat this for all errors on latent variables that are not below the latent variable.
10. Clean up the paths.
   If you do not like where a path has been connected to its variable, use the Select tool, ✂, to click on the path, and then simply click on where it connects to an oval and drag the endpoint.

11. Estimate.
   Click on the Estimate button, ✉️, in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

   Tip: See the tips of [SEM] Example 9 to make creating paths somewhat easier than described above.

   You can open a completed diagram in the Builder by typing
   . webgetsem sem_hcfa1

Reference

Also see
[SEM] sem — Structural equation model estimation command
sem can be used to produce correlations or covariances between exogenous variables. The advantages of using sem over Stata’s correlate command are that you can perform statistical tests on the results and that you can handle missing values in a more elegant way.

To demonstrate these features, we use

```
. use https://www.stata-press.com/data/r16/census13
   (1980 Census data by state)
. describe
Contains data from https://www.stata-press.com/data/r16/census13.dta
obs: 50 1980 Census data by state
vars: 9 2 Dec 2018 14:01

 storage   display value
  variable name   type   format label   variable label
  state         byte   %13.0g state1 State
  brate         int     %10.0g Birth rate
  pop           long   %12.0gc Population
  medage        float   %9.2f Median age
  division      byte   %8.0g division Census division
  region        byte   %-8.0g cenreg Census region
  mrgrate       float   %9.0g Marriage rate
  dvcrate       float   %9.0g Divorce rate
  medagesq      float   %9.0g
```

Sorted by:

See Correlations in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Using sem to obtain correlation matrices
- Fitting the model with the Builder
- Testing correlations with estat stdize and test

Using sem to obtain correlation matrices

We fit the following model:

```
. sem (mrgrate <- starmarchate medagesq) (dvcrate <- starmarchate medagesq)
```

```
   Observed variable equations

   mrgrate <- starmarchate medagesq
   dvcrate <- starmarchate medagesq
```

```
   Residual covariance matrix:
   [2,0] 0.2878
   [1,1] 0.2878
```

See [SEM] Intro 5 for examples.
This model does nothing more than estimate the covariances (correlations), something we could obtain from the `correlate` command by typing

```
. correlate mrgrate dvcrate medage
(obs=50)
```

<table>
<thead>
<tr>
<th></th>
<th>mrgrate</th>
<th>dvcrate</th>
<th>medage</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrgrate</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dvcrate</td>
<td>0.7700</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>medage</td>
<td>-0.0177</td>
<td>-0.2229</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

```
. correlate mrgrate dvcrate medage, covariance
(obs=50)
```

<table>
<thead>
<tr>
<th></th>
<th>mrgrate</th>
<th>dvcrate</th>
<th>medage</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrgrate</td>
<td>.000662</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dvcrate</td>
<td>.000063</td>
<td>1.0e-05</td>
<td></td>
</tr>
<tr>
<td>medage</td>
<td>-.000769</td>
<td>-.001191</td>
<td>2.86775</td>
</tr>
</tbody>
</table>

As explained in Correlations in [SEM] Intro 5, to see results presented as correlations rather than as covariances, we specify `sem`'s standardized option:

```
. sem ( <- mrgrate dvcrate medage), standardized
```

Exogenous variables
- `mrgrate`
- `dvcrate`
- `medage`

Fitting target model:
- Iteration 0: log likelihood = 258.58985
- Iteration 1: log likelihood = 258.58985

Structural equation model
Number of obs = 50
Estimation method = ml
Log likelihood = 258.58985

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Standardized Coef. Std. Err. z P&gt;</td>
</tr>
<tr>
<td>mean(mrgrate)</td>
<td>.7332509 .1593002 4.60 0.000 .4210282 1.045474</td>
</tr>
<tr>
<td>mean(dvcrate)</td>
<td>2.553791 .291922 8.75 0.000 1.981634 3.125947</td>
</tr>
<tr>
<td>mean(medage)</td>
<td>17.62083 1.767749 9.97 0.000 14.15611 21.08556</td>
</tr>
<tr>
<td>var(mrgrate)</td>
<td>1 .            .            .            .</td>
</tr>
<tr>
<td>var(dvcrate)</td>
<td>1 .            .            .            .</td>
</tr>
<tr>
<td>var(medage)</td>
<td>1 .            .            .            .</td>
</tr>
<tr>
<td>cov(mrgrate, dvcrate)</td>
<td>.7699637 .0575805 13.37 0.000 .6571079 .8828195</td>
</tr>
<tr>
<td>cov(mrgrate, medage)</td>
<td>-.0176541 .1413773 -0.12 0.901 -.2947485 .2594403</td>
</tr>
<tr>
<td>cov(dvcrate, medage)</td>
<td>-.222932 .1343929 -1.66 0.097 -.4863373 .0404732</td>
</tr>
</tbody>
</table>

LR test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .
Note:

1. The correlations reported are

<table>
<thead>
<tr>
<th></th>
<th>sem</th>
<th>correlate</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrgrate and dvcrate</td>
<td>0.769937</td>
<td>0.7700</td>
</tr>
<tr>
<td>mrgrate and medage</td>
<td>-0.0176541</td>
<td>-0.0177</td>
</tr>
<tr>
<td>dvcrate and medage</td>
<td>-0.222932</td>
<td>-0.2229</td>
</tr>
</tbody>
</table>

Fitting the model with the Builder

Use the diagram above for reference.

1. Open the dataset.

   In the Command window, type

   . use https://www.stata-press.com/data/r16/census13

2. Open a new Builder diagram.

   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Create the set of observed variables.

   Select the Add observed variables set tool, , and then click in the diagram about halfway down from the top and a quarter of the way in from the left.

   In the resulting dialog box,
   
a. select the Select variables radio button (it may already be selected);
   b. use the Variables control to select the variables in this order: mrgrate, dvcrate, and medage;
   c. select Horizontal in the Orientation control;
   d. select the Distances tab;
   e. select .5 (inch) in the Distance between variables control;
   f. click on OK.

   If you wish, move the set of variables by clicking on any variable and dragging it.

   Be sure you select the observed variables in the order indicated above; otherwise, the instructions below for creating covariances will not be correct.

4. Correlate each pair of variables.

   a. Select the Add covariance tool, .

   b. Click in the top of the mrgrate rectangle, slightly to the right of the center (it will highlight when you hover over it), and drag a path to the top of the dvcrate rectangle, slightly to the left of the center (it will highlight when you can release to connect the covariance).

   c. Click in the top of the dvcrate rectangle, slightly to the right of the center, and drag a path to the top of the medage rectangle, slightly to the left of the center.

   d. Click in the top of the mrgrate rectangle, slightly to the left of the center, and drag a path to the top of the medage rectangle, slightly to the right of the center.
5. Clean up.

If you do not like where a covariance has been connected to its variable, use the Select tool, \[\text{Select tool}\], to click on the covariance, and then simply click on where it connects to an oval and drag the endpoint. You can also change the bow of the covariance by dragging the control point that extends from one end of the selected covariance.


Click on the \textbf{Estimate} button, \[\text{Estimate button}\], in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{SEM estimation options} dialog box.

7. Show standardized estimates.

From the \textit{SEM Builder} menu, select \textbf{View > Standardized estimates}.

You can open a completed diagram in the Builder by typing

\[\texttt{. webgetsem sem_corr}\]

Testing correlations with \texttt{estat stdize} and \texttt{test}

We can test whether the correlations between median age and marriage and divorce rates are equal with \texttt{test} by typing

\[\texttt{. estat stdize: ///}\]
\[\texttt{test _b[/cov(medage,mrgrate)] = _b[/cov(medage,dvcrate)]}\]

We must prefix \texttt{test} with \texttt{estat stdize} because otherwise we would be testing equality of covariances; see \textit{Displaying other results, statistics, and tests (sem and gsem)} in \texttt{[SEM] Intro 7} and see \texttt{[SEM] estat stdize}.

That we refer to the two correlations (covariances) by typing \texttt{_b[/cov(medage,mrgrate)]} and \texttt{_b[/cov(medage,dvcrate)]} is something nobody remembers and that we remind ourselves of by redisplaying \textit{sem} results with the \texttt{coeflegend} option:

\[\texttt{. sem, coeflegend}\]

\begin{verbatim}
Structural equation model
Estimation method = ml
Log likelihood = 258.58985

L R test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .

\end{verbatim}
We can now obtain the test:

```
. estat stdize:
> test _b[/cov(medage,mrgrate)] = _b[/cov(medage,dvcrate)]
( 1) [./cov(mrgrate,medage) - ./cov(dvcrate,medage) = 0
   chi2( 1) = 4.78
   Prob > chi2 = 0.0288
```

Note:
1. We can reject the test at the 5% level.

Also see

[SEM] estat stdize — Test standardized parameters
[SEM] test — Wald test of linear hypotheses
[R] correlate — Correlations of variables
Example 17 — Correlated uniqueness model

Description

To demonstrate a correlated uniqueness model, we use the following summary statistics data:

```
. use https://www.stata-press.com/data/r16/sem_cu1
(Correlated uniqueness)
. ssd describe

obs: 500 Correlated uniqueness
vars: 9 18 Jan 2019 09:34
(_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>par_i</td>
<td>self-report inventory for paranoid</td>
</tr>
<tr>
<td>szt_i</td>
<td>self-report inventory for schizotypal</td>
</tr>
<tr>
<td>szd_i</td>
<td>self-report inventory for schizoid</td>
</tr>
<tr>
<td>par_c</td>
<td>clinical interview rating for paranoid</td>
</tr>
<tr>
<td>szt_c</td>
<td>clinical interview rating for schizotypal</td>
</tr>
<tr>
<td>szd_c</td>
<td>clinical interview rating for schizoid</td>
</tr>
<tr>
<td>par_o</td>
<td>observer rating for paranoid</td>
</tr>
<tr>
<td>szt_o</td>
<td>observer rating for schizotypal</td>
</tr>
<tr>
<td>szd_o</td>
<td>observer rating for schizoid</td>
</tr>
</tbody>
</table>
```

. notes

_dta:
2. Summary statistics represent a sample of 500 patients who were evaluated for three personality disorders using three different methods.
3. The personality disorders include paranoid, schizotypal, and schizoid.
4. The methods of evaluation include a self-report inventory, ratings from a clinical interview, and observational ratings.

See Correlated uniqueness model in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- Fitting the model
- Fitting the model with the Builder

Fitting the model

We fit the following model:

```
.par_i
<table>
<thead>
<tr>
<th>ε1</th>
</tr>
</thead>
</table>
.szt_i
<table>
<thead>
<tr>
<th>ε2</th>
</tr>
</thead>
</table>
.szd_i
<table>
<thead>
<tr>
<th>ε3</th>
</tr>
</thead>
</table>
.par_c
| ε4 |
.szt_c
<table>
<thead>
<tr>
<th>ε5</th>
</tr>
</thead>
</table>
.szd_c
<table>
<thead>
<tr>
<th>ε6</th>
</tr>
</thead>
</table>
.par_o
| ε7 |
.szt_o
<table>
<thead>
<tr>
<th>ε8</th>
</tr>
</thead>
</table>
.szd_o
| ε9 |
```

Endogenous variables

Measurement: par_i par_c par_o szt_i szt_c szt_o szd_i szd_c szd_o

Exogenous variables

Latent: Par Szt Szd

Fitting target model:

Iteration 0: log likelihood = -10210.31 (not concave)
Iteration 1: log likelihood = -10040.188 (not concave)
Iteration 2: log likelihood = -9971.4015
Iteration 3: log likelihood = -9918.0037
Iteration 4: log likelihood = -9883.6368
Iteration 5: log likelihood = -9880.0242
Iteration 6: log likelihood = -9879.9961
Iteration 7: log likelihood = -9879.9961
Example 17 — Correlated uniqueness model

<table>
<thead>
<tr>
<th>Standardized</th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>par_i &lt;- Par</td>
<td>.7119709  .0261858  27.19  0.000  .6606476  .7632941</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>par_c &lt;- Par</td>
<td>.8410183  .0242205  34.72  0.000  .7935469  .8884897</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>par_o &lt;- Par</td>
<td>.7876062  .0237685  33.14  0.000  .7410209  .8341916</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szt_i &lt;- Szt</td>
<td>.7880887  .0202704  38.88  0.000  .7483594  .8278179</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szt_c &lt;- Szt</td>
<td>.7675732  .0244004  31.46  0.000  .7197493  .8153972</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szt_o &lt;- Szt</td>
<td>.8431662  .0181632  46.42  0.000  .807567  .8787653</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szd_i &lt;- Szd</td>
<td>.7692321  .0196626  39.12  0.000  .7306942  .807777</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szd_c &lt;- Szd</td>
<td>.8604596  .0179455  47.95  0.000  .8252871  .8956321</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>szd_o &lt;- Szd</td>
<td>.8715597  .0155875  55.91  0.000  .8410086  .9021107</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{var(e.par_i)} = .4930975  .0372871  .4251739  .5718722 \\
\text{var(e.par_c)} = .2926882  .0407398  .2228049  .3844905 \\
\text{var(e.par_o)} = .3796764  .0374404  .3129503  .4606295 \\
\text{var(e.szt_i)} = .3789163  .0319498  .3211966  .4470082 \\
\text{var(e.szt_c)} = .4108313  .0374582  .3436006  .4912169 \\
\text{var(e.szt_o)} = .2890708  .0306291  .2348623  .3557912 \\
\text{var(e.szd_i)} = .408282  .0302501  .3530966  .4720922 \\
\text{var(e.szd_c)} = .2596093  .0308827  .2056187  .3277766 \\
\text{var(e.szd_o)} = .2403837  .027171  .192616  .2999976 \\
\text{var(Par)} = 1  . .  . \\
\text{var(Szt)} = 1  .  .  . \\
\text{var(Szd)} = 1  .  .  . 
\]
Notes:

1. We use the correlated uniqueness model fit above to analyze a multitrait–multimethod (MTMM) matrix. The MTMM matrix was developed by Campbell and Fiske (1959) to evaluate construct validity of measures. Each trait is measured using different methods, and the correlation matrix produced is used to evaluate whether measures that are related in theory are related in fact (convergent validity) and whether measures that are not intended to be related are not related in fact (discriminant validity).

In this example, the traits are the latent variables Par, Szt, and Szd. The observed variables are the method–trait combinations. The observed traits are the personality disorders paranoid (par), schotypal (szt), and schizoid (szd). The methods used to measure them are self-report (_i), clinical interview (_c), and observer rating (_o). Thus variable par_i is paranoid (par) measured by self-report (_i).

2. Note our use of the covstructure() option, which we abbreviated to covstr(). We used this option instead of cov() to save typing; see Correlated uniqueness model in [SEM] Intro 5.

3. Large values of the factor loadings (path coefficients) indicate convergent validity.

4. Small correlations between latent variables indicate discriminant validity.

Fitting the model with the Builder

Use the diagram above for reference.

There are many ways to draw this diagram. This one produces the cleanest diagram most easily.

1. Open the dataset.

   In the Command window, type

   . use https://www.stata-press.com/data/r16/sem_cu1
2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation.**

3. Change the size of the observed variables’ rectangles.
   a. In the SEM Builder menu, select **Settings > Variables > All observed...**
   b. In the resulting dialog box, change the first size to .44 and click on **OK.**

4. Create the nine measurement variables.
   Select the Add observed variables set tool, ⊕, and then click in the far left of the diagram, about one-third of the way up from the bottom.
   In the resulting dialog box,
   a. select the **Select variables** radio button (it may already be selected);
   b. use the **Variables** control to select the nine variables in this order: \( \text{par}_i, \text{szt}_i, \text{szd}_i, \text{par}_c, \text{szt}_c, \text{szd}_c, \text{par}_o, \text{szt}_o, \text{and szd}_o; \)
   c. select **Horizontal** in the **Orientation control**;
   d. click on **OK.**
   If you wish, move the set of variables by clicking on any variable and dragging it.

Be sure you select the observed variables in the order indicated above; otherwise, the instructions below for creating covariances among the errors will not be correct.

5. Create the set of three latent variables.
   Select the Add latent variables set tool, ☢, and then click above the \( \text{szt}_i \) rectangle, about one-third of the way down from the top of the diagram.
   In the resulting dialog box,
   a. select the **Select variables** radio button (it may already be selected);
   b. type the three latent variable names **Par, Szt, and Szd** into the **Variables** control;
   c. select **Horizontal** in the **Orientation control**;
   d. select the **Distances** tab;
   e. select 1 (inch) from the **Distance between variables** control;
   f. click on **OK.**
   If you wish, move the set of variables by clicking on any variable and dragging it.

We could have forgone specifying the distances between variables and simply dragged each variable where we wished after it was created.

6. Create the factor-loading paths.
   a. Select the Add path tool, −.
   b. Click in the bottom of the **Par** oval (it will highlight when you hover over it), and drag a path to the top of the \( \text{par}_i \) rectangle (it will highlight when you can release to connect the path).
c. Continuing with the tool, create the following paths by clicking first in the bottom of the latent variable and dragging it to the top of the observed (measurement) variable:

\[
\begin{align*}
Szt & \rightarrow szt_i \\
Szd & \rightarrow szd_i \\
Par & \rightarrow par_c \\
Szt & \rightarrow szt_c \\
Szd & \rightarrow szd_c \\
Par & \rightarrow par_o \\
Szt & \rightarrow szt_o \\
Szd & \rightarrow szd_o
\end{align*}
\]

7. Clean up the direction of the errors.

We want all the errors to be below the measurement variables.

a. Choose the Select tool, .

b. Click on the rectangle of any measurement variable whose associated error is not below it.

c. Click on one of the Error rotation buttons, , in the Contextual Toolbar until the error is below the measurement variable.

Repeat this for all errors that are not below the measurement variables.

8. Correlate the errors within the self-report, clinical interview, and observer rating groups.

a. Select the Add covariance tool, .

b. Click in the \( \epsilon_2 \) circle (it will highlight when you hover over it), and drag a covariance to the \( \epsilon_1 \) circle (it will highlight when you can release to connect the covariance).

c. Continue with the Add covariance tool, , to create eight more covariances by clicking the first-listed error and dragging it to the second-listed error.

\[
\begin{align*}
\epsilon_3 & \rightarrow \epsilon_2 \\
\epsilon_3 & \rightarrow \epsilon_1 \\
\epsilon_5 & \rightarrow \epsilon_4 \\
\epsilon_6 & \rightarrow \epsilon_5 \\
\epsilon_6 & \rightarrow \epsilon_4 \\
\epsilon_8 & \rightarrow \epsilon_7 \\
\epsilon_9 & \rightarrow \epsilon_8 \\
\epsilon_9 & \rightarrow \epsilon_7
\end{align*}
\]

The order in which we create the covariances is unimportant. We dragged each covariance from right to left because the bow of the covariance is outward when we drag in a clockwise direction and inward when we drag in a counterclockwise direction. Had we connected the opposite way, we would have needed to use the Contextual Toolbar to mirror the bow of the covariances.

9. Correlate the latent factors.

a. Select the Add covariance tool, .

b. Click in the Par oval and drag a covariance to the Szt oval.

c. Click in the Szt oval and drag a covariance to the Szd oval.

d. Click in the Par oval and drag a covariance to the Szd oval.
10. Clean up.
   If you do not like where a covariance has been connected to its variable, use the Select tool, , to click on the covariance, and then simply click on where it connects to an oval and drag the endpoint. You can also change the bow of the covariance by dragging the control point that extends from one end of the selected covariance.

11. Estimate.
   Click on the Estimate button, , in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

12. Show standardized estimates.
   From the SEM Builder menu, select View > Standardized estimates.

Reference

Also see
[SEM] sem — Structural equation model estimation command
[SEM] sem and gsem option covstructure( ) — Specifying covariance restrictions
Example 18 — Latent growth model

Description

To demonstrate a latent growth model, we use the following data:

```
. use https://www.stata-press.com/data/r16/sem_lcm
. describe
Contains data from https://www.stata-press.com/data/r16/sem_lcm.dta
    obs:     359   25 May 2018 11:08
    vars:    4
(  _dta has notes)

Variable                      Storage   Display  Value  Label
---                           ------     -------  ------  -----
    lncrime0: ln(crime rate) in Jan & Feb
    lncrime1: ln(crime rate) in Mar & Apr
    lncrime2: ln(crime rate) in May & Jun
    lncrime3: ln(crime rate) in Jul & Aug
```

Sorted by:

```
. notes
_dta:
```

See _Latent growth models_ in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the model
- Fitting the model with the Builder
Fitting the model

We fit the following model:

```
sem (Incrime0 <- Intercept Slope _cons)
>   (Incrime1 <- Intercept Slope _cons)
>   (Incrime2 <- Intercept Slope _cons)
>   (Incrime3 <- Intercept Slope _cons),
   latent(Intercept Slope)
   var(e.Incrime0 e.Incrime1 e.Incrime2 e.Incrime3)
   means(Intercept Slope)
```

Endogenous variables
Measurement: Incrime0 Incrime1 Incrime2 Incrime3
Exogenous variables
Latent: Intercept Slope

Fitting target model:
Iteration 0:  log likelihood = -1034.1038
Iteration 1:  log likelihood = -1033.9044
Iteration 2:  log likelihood = -1033.9037
Iteration 3:  log likelihood = -1033.9037

Structural equation model
Number of obs = 359
Estimation method = ml
Log likelihood = -1033.9037

( 1) [Incrime0]Intercept = 1
( 2) [Incrime1]Intercept = 1
( 3) [Incrime1]Slope = 1
( 4) [Incrime2]Intercept = 1
( 5) [Incrime2]Slope = 2
( 6) [Incrime3]Intercept = 1
( 7) [Incrime3]Slope = 3
( 8) [\]var(e.Incrime0) - [\]var(e.Incrime3) = 0
( 9) [\]var(e.Incrime1) - [\]var(e.Incrime3) = 0
(10) [\]var(e.Incrime2) - [\]var(e.Incrime3) = 0
(11) [Incrime0]_cons = 0
(12) [Incrime1]_cons = 0
(13) [Incrime2]_cons = 0
(14) [Incrime3]_cons = 0
### Example 18 — Latent growth model

| Measurement | OIM | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------|-----|-------|-----------|-----|-----|----------------------|
| lncrime0    |     |       |           |     |     |                      |
| Intercept   | 1   | (constrained) |
| _cons       | 0   | (constrained) |
| lncrime1    |     |       |           |     |     |                      |
| Intercept   | 1   | (constrained) |
| Slope       | 1   | (constrained) |
| _cons       | 0   | (constrained) |
| lncrime2    |     |       |           |     |     |                      |
| Intercept   | 1   | (constrained) |
| Slope       | 2   | (constrained) |
| _cons       | 0   | (constrained) |
| lncrime3    |     |       |           |     |     |                      |
| Intercept   | 1   | (constrained) |
| Slope       | 3   | (constrained) |
| _cons       | 0   | (constrained) |

| mean(Inter−t) | 5.337915 | 0.0407501 | 130.99 | 0.000 | 5.258047 | 5.417784 |
| mean(Slope)   | 0.1426952 | 0.0104574 | 13.65  | 0.000 | 0.1221992 | 0.1631912 |

| var(e.lncr−0) | 0.0981956 | 0.0051826 | 0.0885457 | 0.1088972 |
| var(e.lncr−1) | 0.0981956 | 0.0051826 | 0.0885457 | 0.1088972 |
| var(e.lncr−2) | 0.0981956 | 0.0051826 | 0.0885457 | 0.1088972 |
| var(e.lncr−3) | 0.0981956 | 0.0051826 | 0.0885457 | 0.1088972 |
| var(Inter−t)  | 0.527409  | 0.0446436 | 0.4467822 | 0.6225858 |
| var(Slope)    | 0.0196198 | 0.0031082 | 0.0143829 | 0.0267635 |

| cov(Inter−t, Slope) | -0.034316 | 0.0088848 | -3.86 | 0.000 | -0.0517298 | -0.0169022 |

LR test of model vs. saturated: chi2(8) = 16.25, Prob > chi2 = 0.0390

Notes:

1. In this example, we have repeated measures of the crime rate in 1995. We will assume that the underlying rate grows linearly.

2. As explained in *Latent growth models* in [SEM] Intro 5, we assume

   \[ \text{lncrime}_i = \text{Intercept} + i \times \text{Slope} \]

3. *sem* does not usually report the means of latent exogenous variables because *sem* automatically includes the identifying constraint that the means are 0; see *How sem (gsem) solves the problem for you* in [SEM] Intro 4 and see *Default normalization constraints* in [SEM] sem.

   In this case, *sem* did not constrain the means to be 0 because we specified *sem*'s `means()` option. In particular, we specified `means(Intercept Slope)`, which said not to constrain the means of those two exogenous latent variables and to report the estimated result.

   Our model was identified even without the usual 0 constraints on Intercept and Slope because we specified enough other constraints.

4. We estimate the Intercept to have mean 5.34 and the mean Slope to be 0.14 per two months. Remember, we have measured crime rates as log base e crime rates.
5. It might help some to think of this as a mixed model:

\[
\begin{align*}
&\text{generate id = _n} \\
&\text{reshape long lncrime, i(id) j(year)} \\
&\text{mixed lncrime year || id:year, cov(unstructured) mle var}
\end{align*}
\]

The mean Intercept and Slope are what mixed would refer to as the coefficients in the fixed-effects part of the model.

Acock (2013, chap. 4) discusses the use of sem to fit latent growth-curve models in more detail. Acock demonstrates extensions to the basic model we fit here, such as including time-varying and time-invariant covariates in the model.

**Fitting the model with the Builder**

Use the diagram above for reference.

1. Open the dataset.
   
   In the Command window, type
   
   \[
   \text{. use https://www.stata-press.com/data/r16/sem_lcm}
   \]

2. Open a new Builder diagram.
   
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Create the four repeated measurements for log crime rate.
   
   Select the Add observed variables set tool, and then click in the diagram about one-fourth of the way in from the left and about one-third of the way up from the bottom.
   
   In the resulting dialog box,
   
   a. select the Select variables radio button (it may already be selected);
   b. use the Variables control to select the variables in this order: lncrime0, lncrime1, lncrime2, and lncrime3;
   c. select Horizontal in the Orientation control;
   d. select the Distances tab;
   e. select .25 (inch) from the Distance between variables control;
   f. click on OK.
   
   If you wish, move the set of variables by clicking on any variable and dragging it.

4. Create the two latent variables.
   
   Select the Add latent variables set tool, and then click in the diagram about one-third of the way down from the top and above the left side of the lncrime0 rectangle.
   
   In the resulting dialog box,
   
   a. select the Select variables radio button (it may already be selected);
   b. type the latent variable names Intercept and Slope into the Variables control;
   c. select Horizontal in the Orientation control;
   d. select the Distances tab;
e. select 1 (inch) from the Distance between variables control;
f. click on OK.

If you wish, move the set of variables by clicking on any variable and dragging it.

We could have forgone specifying the distances between variables and simply dragged each variable where we wished after it was created.

5. Create the factor-loading paths.

a. Select the Add path tool, →.
b. Click in the bottom-left quadrant of the Intercept oval (it will highlight when you hover over it), and drag a path to the top of the lncrime0 rectangle (it will highlight when you can release to connect the path).
c. Continuing with the → tool, create the following paths by clicking first in the bottom of the latent variable and dragging it to the top of the observed (measurement) variable:

\[
\begin{align*}
\text{Intercept} & \rightarrow \text{lncrime1} \\
\text{Intercept} & \rightarrow \text{lncrime2} \\
\text{Intercept} & \rightarrow \text{lncrime3} \\
\text{Slope} & \rightarrow \text{lncrime0} \\
\text{Slope} & \rightarrow \text{lncrime1} \\
\text{Slope} & \rightarrow \text{lncrime2} \\
\text{Slope} & \rightarrow \text{lncrime3}
\end{align*}
\]

6. Clean up the direction of the errors.

We want all the errors to be below the measurement variables.

a. Choose the Select tool, 🗼.
b. Click on the rectangle of any measurement variable whose associated error is not below it.
c. Click on one of the Error rotation buttons, ← →, in the Contextual Toolbar until the error is below the measurement variable.

Repeat this for all errors that are not below the measurement variables.

7. Create the covariance between Intercept and Slope.

a. Select the Add covariance tool, ⬅️.
b. Click in the top-right quadrant of the Intercept oval, and drag a covariance to the top left of the Slope oval.

8. Clean up paths and covariance.

If you do not like where a path has been connected to its variables, use the Select tool, 🗼, to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint. Similarly, you can change where the covariance connects to the latent variables by clicking on the covariance and dragging the endpoint. You can also change the bow of the covariance by clicking on the covariance and dragging the control point that extends from one end of the selected covariance.

9. Constrain the intercepts of the measurements to 0.

a. Choose the Select tool, 🗼.
b. Click on the rectangle for lncrime0. In the Contextual Toolbar, type 0 in the box and press Enter.

c. Repeat this process to add the 0 constraint on the intercept for lncrime1, lncrime2, and lncrime3.

10. Set constraints on the paths from Intercept to the measurements.

   a. Continue with the Select tool.

   b. Click on the path from Intercept to lncrime0. In the Contextual Toolbar, type 1 in the box and press Enter.

   c. Repeat this process to add the 1 constraint on the following paths:

      Intercept -> lncrime1
      Intercept -> lncrime2
      Intercept -> lncrime3

11. Set constraints on the paths from Slope to the measurements.

   a. Continue with the Select tool.

   b. Click on the path from Slope to lncrime0. In the Contextual Toolbar, type 0 in the box and press Enter.

   c. Click on the path from Slope to lncrime1. In the Contextual Toolbar, type 1 in the box and press Enter.

   d. Click on the path from Slope to lncrime2. In the Contextual Toolbar, type 2 in the box and press Enter.

   e. Click on the path from Slope to lncrime3. In the Contextual Toolbar, type 3 in the box and press Enter.

12. Set equality constraints on the error variances.

   a. Continue with the Select tool.

   b. Click in the ϵ1 circle, which is the error term for lncrime0. In the Contextual Toolbar, type var in the box and press Enter.

   c. Repeat this process to add the var constraint on the three remaining error variances: ϵ2, ϵ3, and ϵ4.

13. Clean up placement of the constraints.

    From the SEM Builder menu, select Settings > Connections > Paths....

    In the resulting dialog box, do the following:

    a. Click on the Results tab.

    b. Click on the Result 1... button at the bottom left.

    c. In the Appearance of result 1 - paths dialog box that opens, choose 20 (%) in the Distance between nodes control.

    d. Click on OK on the Appearance of result 1 - paths dialog box.

    e. Click on OK on the Connection settings - paths dialog box.
14. Specify that the means of Intercept and Slope are to be estimated.
   a. Choose the Select tool, ![Select tool](select_icon.png).
   b. Double-click on the Intercept oval.
   c. In the resulting dialog box, check the Estimate mean box.
   d. Click on OK.
   e. In the same manner, double-click on Slope and check the Estimate mean box.

15. Estimate.
   Click on the Estimate button, ![Estimate button](estimate_icon.png), in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

You can open a completed diagram in the Builder by typing

```plaintext
.webgetsem sem_lcm
```

**Reference**


**Also see**

[SEM] sem — Structural equation model estimation command
Example 19 — Creating multiple-group summary statistics data

Description

The data analyzed in [SEM] Example 20 are summary statistics data (SSD) and contain summary statistics on two groups of subjects, those from grade 4 and those from grade 5. Below we show how we created this summary statistics dataset.

See [SEM] Intro 11 for background on SSD.

Remarks and examples

See [SEM] Example 2 for creating a single-group dataset from published covariances. In this example, we will create a two-group dataset from published correlations, standard deviations, and means.

Marsh and Hocevar (1985) publish lots of SSD, of which we will enter the data for students in grade 4 and grade 5 found on pages 579–581. In that source, the authors published the correlations, standard deviations, and means of their variables.

We will set the data for the first group, declare that we have groups and wish to add another, and set the data for the second group.

Starting with the first group, we will issue the following commands:

```
    . ssp init variable names
    . ssp set obs values
    . ssp set means values
    . ssp set sd values
    . ssp set corr values
```

We will first set the end-of-line delimiter to a semicolon because we are going to have some long lines. We will be entering SSD for 16 variables!

```
    . #delimit ;
    delimiter now ;
    . ssp init phyab1 phyab2 phyab3 phyab4
    > appear1 appear2 appear3 appear4
    > peerrel1 peerrel2 peerrel3 peerrel4
    > parrel1 parrel2 parrel3 parrel4 ;
```

Summary statistics data initialized. Next use, in any order,

- **SSD set observations** (required)
  - It is best to do this first.
- **SSD set means** (optional)
  - Default setting is 0.
- **SSD set variances** or **SSD set sd** (optional)
  - Use this only if you have set or will set correlations and, even then, this is optional but highly recommended. Default setting is 1.
- **SSD set covariances** or **SSD set correlations** (required)
We have now entered the data for the first group, and ssd reports that we have a fully set dataset.
Next we are going to add a second group by typing

```
    . ssd addgroup grade
    (new group grade==2 added)
```

The `ssd set` commands now modify the new group `grade==2`. If you need to modify data for `grade==1`, place a `1` right after the `set`. For example,

```
    . ssd set 1 means ...
```

would modify the means for group `grade==1`.

The `ssd set` command now modifies the new group `grade==2`. If we needed to modify data for `grade==1`, we would place a `1` right after the `set`. For example,

```
    . ssd set 1 means ...
```

We are not modifying data; however, we are now adding data for the second group. The procedure for entering the second group is the same as the procedure for entering the first group:

```
    . ssd set obs values
    . ssd set means values
    . ssd set sd values
    . ssd set corr values
```

We do that below.

```
    . #delimit ;
    delimiter now ;
    . ssd set obs 251 ;
    (value set for group grade==2)
    Status for group grade==2:
    observations: set
    means: unset
    variances or sd: unset
    covariances or correlations: unset (required to be set)
    . ssd set corr
    >   1.0 \ 
    >   .31 1.0 \ 
    >   .52 .45 1.0 \ 
    >   .54 .46 .70 1.0 \ 
    >   .15 .33 .22 .21 1.0 \ 
    >   .14 .28 .21 .13 .72 1.0 \ 
    >   .16 .32 .35 .31 .59 .56 1.0 \ 
    >   .23 .29 .43 .36 .55 .51 .65 1.0 \ 
    >   .24 .13 .24 .23 .25 .24 .24 .30 1.0 \ 
    >   .19 .26 .22 .18 .34 .37 .36 .32 .38 1.0 \ 
    >   .16 .24 .36 .30 .33 .29 .44 .51 .47 .50 1.0 \ 
    >   .16 .21 .35 .24 .31 .33 .41 .39 .47 .47 .55 1.0 \ 
    >   .08 .18 .09 .12 .19 .24 .08 .21 .21 .19 .19 .20 1.0 \ 
    >   .01 -.01 .03 .02 .10 .13 .03 .05 .26 .17 .23 .26 .33 1.0 \ 
    >   .06 .19 .22 .22 .23 .24 .20 .26 .16 .23 .38 .24 .42 .40 1.0 \ 
    >   .04 .17 .10 .07 .26 .24 .12 .26 .16 .22 .32 .17 .42 .42 .65 1.0 ;
    (values set for group grade==2)
    Status for group grade==2:
    observations: set
    means: unset
    variances or sd: unset
    covariances or correlations: set
```
Example 19 — Creating multiple-group summary statistics data

. ssd set sd 1.84 1.94 2.07 1.82 2.34 2.61 2.48 2.34
> 1.71 1.93 2.18 1.94 1.31 1.57 1.57 1.47 ;
(values set for group grade==2)
Status for group grade==2:
observations: set
means: unset
variances or sd: set
covariances or correlations: set

. ssd set means 8.20 8.23 8.17 8.56 7.41 7.00 7.17 7.40
> 8.81 7.94 7.52 8.29 9.35 9.13 8.67 9.00 ;
(values set for group grade==2)
Status for group grade==2:
observations: set
means: set
variances or sd: set
covariances or correlations: set

. #delimit cr
delimiter now cr

We could stop here and save the data in a Stata dataset. We might type

. save sem_2fmmby

However, we intend to use these data as an example in this manual and online. Here is what you would see if you typed ssd describe:

. ssd describe
Summary statistics data
obs: 385
vars: 16

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>phyab1</td>
<td></td>
</tr>
<tr>
<td>phyab2</td>
<td></td>
</tr>
<tr>
<td>phyab3</td>
<td></td>
</tr>
<tr>
<td>phyab4</td>
<td></td>
</tr>
<tr>
<td>appear1</td>
<td></td>
</tr>
<tr>
<td>appear2</td>
<td></td>
</tr>
<tr>
<td>appear3</td>
<td></td>
</tr>
<tr>
<td>appear4</td>
<td></td>
</tr>
<tr>
<td>peerrel11</td>
<td></td>
</tr>
<tr>
<td>peerrel12</td>
<td></td>
</tr>
<tr>
<td>peerrel13</td>
<td></td>
</tr>
<tr>
<td>peerrel14</td>
<td></td>
</tr>
<tr>
<td>parrel11</td>
<td></td>
</tr>
<tr>
<td>parrel12</td>
<td></td>
</tr>
<tr>
<td>parrel13</td>
<td></td>
</tr>
<tr>
<td>parrel14</td>
<td></td>
</tr>
</tbody>
</table>

Group variable: grade (2 groups)
Obs. by group: 134, 251
We are going to label these data so that `ssd describe` can provide more information:

`.label data "two-factor CFA"
.label var phyab1 "Physical ability 1"
.label var phyab2 "Physical ability 2"
.label var phyab3 "Physical ability 3"
.label var phyab4 "Physical ability 4"
.label var appear1 "Appearance 1"
.label var appear2 "Appearance 2"
.label var appear3 "Appearance 3"
.label var appear4 "Appearance 4"
.label var peerrel1 "Relationship w/ peers 1"
.label var peerrel2 "Relationship w/ peers 2"
.label var peerrel3 "Relationship w/ peers 3"
.label var peerrel4 "Relationship w/ peers 4"
.label var parrel1 "Relationship w/ parent 1"
.label var parrel2 "Relationship w/ parent 2"
.label var parrel3 "Relationship w/ parent 3"
.label var parrel4 "Relationship w/ parent 4"
#.delimit ;
#delimit cr
#delimit ;
#delimit cr
#delimit ;
#delimit cr
#delimit ;
#delimit cr

We would now save the dataset.

To see `ssd describe`'s output with the data labeled, see [SEM] Example 20.

Reference


Also see

[SEM] Example 20 — Two-factor measurement model by group
[SEM] ssd — Making summary statistics data (sem only)
Example 20 — Two-factor measurement model by group

Description

Below we demonstrate sem’s `group()` option, which allows fitting models in which path coefficients and covariances differ across groups of the data, such as for males and females. We use the following data:

```
. use https://www.stata-press.com/data/r16/sem_2fmmby
(two-factor CFA)
. ssd describe
```

Summary statistics data from
https://www.stata-press.com/data/r16/sem_2fmmby.dta

| obs: 385 | two-factor CFA |
| vats: 16 | 25 May 2018 11:11 |
| (_dta has notes) |

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>phyab1</td>
<td>Physical ability 1</td>
</tr>
<tr>
<td>phyab2</td>
<td>Physical ability 2</td>
</tr>
<tr>
<td>phyab3</td>
<td>Physical ability 3</td>
</tr>
<tr>
<td>phyab4</td>
<td>Physical ability 4</td>
</tr>
<tr>
<td>appear1</td>
<td>Appearance 1</td>
</tr>
<tr>
<td>appear2</td>
<td>Appearance 2</td>
</tr>
<tr>
<td>appear3</td>
<td>Appearance 3</td>
</tr>
<tr>
<td>appear4</td>
<td>Appearance 4</td>
</tr>
<tr>
<td>peerrel1</td>
<td>Relationship w/ peers 1</td>
</tr>
<tr>
<td>peerrel2</td>
<td>Relationship w/ peers 2</td>
</tr>
<tr>
<td>peerrel3</td>
<td>Relationship w/ peers 3</td>
</tr>
<tr>
<td>peerrel4</td>
<td>Relationship w/ peers 4</td>
</tr>
<tr>
<td>parrel1</td>
<td>Relationship w/ parent 1</td>
</tr>
<tr>
<td>parrel2</td>
<td>Relationship w/ parent 2</td>
</tr>
<tr>
<td>parrel3</td>
<td>Relationship w/ parent 3</td>
</tr>
<tr>
<td>parrel4</td>
<td>Relationship w/ parent 4</td>
</tr>
</tbody>
</table>

Group variable: grade (2 groups)
Obs. by group: 134, 251

```
. notes
_dta:
2. Summary statistics based on 134 students in grade 4 and 251 students in grade 5 from Sydney, Australia.
3. Group 1 is grade 4, group 2 is grade 5.
4. Data collected using the Self-Description Questionnaire and includes sixteen subscales designed to measure nonacademic status: four intended to measure physical ability, four intended to measure physical appearance, four intended to measure relations with peers, and four intended to measure relations with parents.
```
Remarks and examples

Remarks are presented under the following headings:

Background
Fitting the model with all the data
Fitting the model with the group() option
Fitting the model with the Builder

Background

See [SEM] Intro 6 for background on *sem’s* group() option.

We will fit the model

\[
\begin{align*}
\varepsilon_1 & \rightarrow \text{peerrel1} \\
\varepsilon_2 & \rightarrow \text{peerrel2} \\
\varepsilon_3 & \rightarrow \text{peerrel3} \\
\varepsilon_4 & \rightarrow \text{peerrel4} \\
\varepsilon_5 & \rightarrow \text{parrel1} \\
\varepsilon_6 & \rightarrow \text{parrel2} \\
\varepsilon_7 & \rightarrow \text{parrel3} \\
\varepsilon_8 & \rightarrow \text{parrel4}
\end{align*}
\]

which in command syntax can be written as

\[
\text{(Peer} \rightarrow \text{peerrel1 peerrel2 peerrel3 peerrel4)} \quad /// \quad \text{(Par} \rightarrow \text{parrel1 parrel2 parrel3 parrel4)}
\]

We are using the same data used in [SEM] Example 15, but we are using more of the data and fitting a different model. To remind you, those data were collected from students in grade 5. The dataset we are using, however, has data for students from grade 4 and from grade 5, which was created in [SEM] Example 19. We have the following observed variables:

1. Four measures of physical ability.
2. Four measures of appearance.
3. Four measures of quality of relationship with peers.
4. Four measures of quality of relationship with parents.

In this example, we will consider solely the measurement problem, and include only the measurement variables for the two kinds of relationship quality. We are going to treat quality of relationship with peers as measures of underlying factor Peer and quality of relationship with parents as measures of underlying factor Par.

Below we will

1. Fit the model with all the data. This amounts to assuming that the students in grades 4 and 5 are identical in terms of this measurement problem.
2. Fit the model with *sem’s* group() option, which will constrain some parameters to be the same for students in grades 4 and 5 and leave free of constraint the others.
Fitting the model with all the data

Throughout this example, we want you to appreciate that we are using SSD and that matters not at all. Not one command would have a different syntax or option, or produce a different result, if we had the real data.

We begin by fitting the model with all the data:

```
. sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4) 
   > (Par    -> parrel1 parrel2 parrel3 parrel4) 

Endogenous variables
Measurement: peerrel1 peerrel2 peerrel3 peerrel4 parrel1 parrel2 parrel3 parrel4
Exogenous variables
Latent:       Peer Par
```

Fitting target model:

```
Interaction 0:  log likelihood =  -5559.545
Iteration 1:   log likelihood =  -5558.609
Iteration 2:   log likelihood =  -5558.6017
Iteration 3:   log likelihood =  -5558.6017
```
Example 20 — Two-factor measurement model by group

<table>
<thead>
<tr>
<th>Structural equation model</th>
<th>Number of obs = 385</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimation method = ml</td>
<td>-5558.6017</td>
</tr>
<tr>
<td>Log likelihood</td>
<td>(1) [peerrel1]Peer = 1</td>
</tr>
<tr>
<td></td>
<td>(2) [parrel1]Par = 1</td>
</tr>
</tbody>
</table>

| Measurement | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------------|-------|-----------|-------|------|----------------------|
| peerrel1    |       |           |       |      |                      |
| Peer        | 1     | (constrained) | 8.681221 | .0937197 | 92.63 | 0.000 | 8.497534 | 8.864908 |
| _cons       | 8.832121 | .0937197 | 92.63 | 0.000 | 8.497534 | 8.864908 |
| peerrel2    |       |           |       |      |                      |
| Peer        | 1.113865 | .09796 | 11.37 | 0.000 | .9218666 | 1.305863 |
| _cons       | 7.828623 | .1037547 | 75.45 | 0.000 | 7.625268 | 8.031979 |
| peerrel3    |       |           |       |      |                      |
| Peer        | 1.42191 | .114341 | 12.44 | 0.000 | 1.197806 | 1.646014 |
| _cons       | 7.359896 | .1149905 | 64.00 | 0.000 | 7.134519 | 7.585273 |
| peerrel4    |       |           |       |      |                      |
| Peer        | 1.204146 | .0983865 | 12.24 | 0.000 | 1.011312 | 1.39698 |
| _cons       | 8.150779 | .1023467 | 79.64 | 0.000 | 7.950183 | 8.351375 |
| parrel1     |       |           |       |      |                      |
| Par         | 1     | (constrained) | 9.339558 | .0648742 | 143.96 | 0.000 | 9.212407 | 9.46671 |
| _cons       | 9.339558 | .0648742 | 143.96 | 0.000 | 9.212407 | 9.46671 |
| parrel2     |       |           |       |      |                      |
| Par         | 1.112383 | .1378687 | 8.07 | 0.000 | .8421655 | 1.382601 |
| _cons       | 9.220494 | .0742356 | 124.21 | 0.000 | 9.074994 | 9.365993 |
| parrel3     |       |           |       |      |                      |
| Par         | 2.037924 | .204617 | 9.96 | 0.000 | 1.636882 | 2.438966 |
| _cons       | 8.676961 | .088927 | 97.57 | 0.000 | 8.502667 | 8.851255 |
| parrel4     |       |           |       |      |                      |
| Par         | 1.52253 | .1536886 | 9.91 | 0.000 | 1.221309 | 1.82375 |
| _cons       | 9.045247 | .0722358 | 125.22 | 0.000 | 8.903667 | 9.186826 |
| var(e.peer-1) | 1.809309 | .1596546 | 1.521956 | 2.150916 |
| var(e.peer-2) | 2.193804 | .194494 | 1.843884 | 2.610129 |
| var(e.peer-3) | 1.911874 | .214104 | 1.535099 | 2.381126 |
| var(e.peer-4) | 1.753037 | .1749613 | 1.441575 | 2.131792 |
| var(e.parr-1) | 1.120333 | .0899209 | 9.572541 | 1.311193 |
| var(e.parr-2) | 1.503003 | .1200739 | 1.285162 | 1.757769 |
| var(e.parr-3) | .9680081 | .1419777 | 0.7261617 | 1.290401 |
| var(e.parr-4) | .8498834 | .0933687 | 0.685245 | 1.054078 |
| var(Peer)    | 1.572294 | .2265704 | 1.186904 | 2.082822 |
| var(Par)     | 5.000002 | .093189 | 54.69938 | 1.720470 |
| cov(Peer,Par) | .4226706 | .0725253 | 5.83 | 0.000 | .2805236 | .5648176 |

LR test of model vs. saturated: chi2(19) = 28.19, Prob > chi2 = 0.0798

Note:

1. We are using SSD with data for two separate groups. There is no hint of that in the output above because sem combined the summary statistics and produced overall results just as if we had the real data.
Example 20 — Two-factor measurement model by group

Fitting the model with the group() option

```plaintext
. sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4)
> (Par -> parrel1 parrel2 parrel3 parrel4), group(grade)
```

Endogenous variables

Measurement: peerrel1 peerrel2 peerrel3 peerrel4 parrel1 parrel2 parrel3 parrel4

Exogenous variables

Latent: Peer Par

Fitting target model:

Iteration 0:  log likelihood =  -13049.77  (not concave)
Iteration 1:  log likelihood =  -10819.682 (not concave)
Iteration 2:  log likelihood = -8873.4568 (not concave)
Iteration 3:  log likelihood = -6119.7114 (not concave)
Iteration 4:  log likelihood = -5949.354 (not concave)
Iteration 5:  log likelihood = -5775.6085 (not concave)
Iteration 6:  log likelihood = -5713.9178 (not concave)
Iteration 7:  log likelihood = -5638.1208 (not concave)
Iteration 8:  log likelihood = -5616.6335 (not concave)
Iteration 9:  log likelihood = -5595.7507 (not concave)
Iteration 10: log likelihood = -5589.9802 (not concave)
Iteration 11: log likelihood = -5578.8701 (not concave)
Iteration 12: log likelihood = -5574.0162 (not concave)
Iteration 13: log likelihood = -5568.0786
Iteration 14: log likelihood = -5551.7349
Iteration 15: log likelihood = -5544.0052
Iteration 16: log likelihood = -5542.7113
Iteration 17: log likelihood = -5542.6775
Iteration 18: log likelihood = -5542.6774

Structural equation model

Number of obs = 385
Grouping variable = grade
Number of groups = 2
Estimation method = ml
Log likelihood = -5542.6774

( 1)  [peerrel1]1bn.grade#c.Peer  =  1
( 2)  [peerrel2]1bn.grade#c.Peer  -  [peerrel2]2.grade#c.Peer  =  0
( 3)  [peerrel3]1bn.grade#c.Peer  -  [peerrel3]2.grade#c.Peer  =  0
( 4)  [peerrel4]1bn.grade#c.Peer  -  [peerrel4]2.grade#c.Peer  =  0
( 5)  [parrel1]1bn.grade#c.Par  =  1
( 6)  [parrel2]1bn.grade#c.Par  -  [parrel2]2.grade#c.Par  =  0
( 7)  [parrel3]1bn.grade#c.Par  -  [parrel3]2.grade#c.Par  =  0
( 8)  [parrel4]1bn.grade#c.Par  -  [parrel4]2.grade#c.Par  =  0
( 9)  [peerrel1]2.grade  -  [peerrel1]2.grade  =  0
(10)  [peerrel2]2.grade  -  [peerrel2]2.grade  =  0
(11)  [peerrel3]2.grade  -  [peerrel3]2.grade  =  0
(12)  [peerrel4]2.grade  -  [peerrel4]2.grade  =  0
(13)  [parrel1]2.grade  -  [parrel1]2.grade  =  0
(14)  [parrel2]2.grade  -  [parrel2]2.grade  =  0
(15)  [parrel3]2.grade  -  [parrel3]2.grade  =  0
(16)  [parrel4]2.grade  -  [parrel4]2.grade  =  0
(17)  [peerrel1]2.grade#c.Peer  =  1
(18)  [parrel1]2.grade#c.Par  =  1
(19)  [/]mean(Peer)#1bn.grade  =  0
(20)  [/]mean(Par)#1bn.grade  =  0
| Group | Measurement | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-------------|-------|-----------|-------|-------|----------------------|
|       | peerre1     | 1 (constrained) | 8.466539 | .1473448 | 57.46 | 0.000 | 8.177748 8.755329 |
|       | peerre2     | 1.109234 | .0975279 | 11.37 | 0.000 | .9180833 1.300385 |
|       | peerre3     | 1.409361 | .1138314 | 12.38 | 0.000 | 1.186256 1.632467 |
|       | peerre4     | 1.195982 | .0980272 | 12.20 | 0.000 | 1.003852 1.388112 |
|       | parre1      | 1 (constrained) | 9.368654 | .0819489 | 114.32 | 0.000 | 9.208037 9.529271 |
|       | parre2      | 1.104355 | .1369365 | 8.06 | 0.000 | .8359649 1.372746 |
|       | parre3      | 2.058599 | .2060583 | 9.99 | 0.000 | 1.654723 2.462457 |
|       | parre4      | 1.526706 | .1528468 | 9.83 | 0.000 | 1.222424 1.830987 |
|       | mean(Peer)  | 0 (constrained) | 0 | | | |
|       | mean(Par)   | 0 (constrained) | 0 | | | |
|       | var(e.peer-1) | 1.824193 | .2739446 | 6.65 | 0.000 | 1.359074 2.448489 |
|       | var(e.peer-2) | 2.236974 | .3310875 | 6.74 | 0.000 | 1.673699 2.898817 |
|       | var(e.peer-3) | 1.907009 | .3383293 | 5.65 | 0.000 | 1.346908 2.700023 |
|       | var(e.peer-4) | 1.639881 | .272764 | 6.01 | 0.000 | 1.18367 2.271925 |
|       | var(e.parr-1) | 1.469121 | .1302489 | 11.26 | 0.000 | .7425488 2.259067 |
|       | var(e.parr-2) | .9683788 | .133192 | 7.29 | 0.000 | .7395628 1.268012 |
|       | var(e.parr-3) | .8377567 | .1986089 | 4.24 | 0.000 | .526407 1.333258 |
|       | var(e.parr-4) | .8343032 | .1384649 | 6.01 | 0.000 | .6026352 1.15503 |
|       | var(Peer)   | 2.039297 | .3784544 | 5.41 | 0.000 | 1.41747 2.933912 |
|       | var(Par)    | .4492996 | .1011565 | 4.45 | 0.000 | .2889976 .6985183 |
|       | cov(Peer,Par) | .5012091 | .1193333 | 4.20 | 0.000 | .2673201 .7350982 |
### Example 20 — Two-factor measurement model by group

**Group**: 2  
**Number of obs**: 251

| Measurement | GIM | Coef.  | Std. Err. | z      | P>|z| | [95% Conf. Interval] |
|-------------|-----|--------|-----------|--------|------|----------------------|
| peerrel1    | Peer | 1 (constrained) | 8.466539 | .1473448 | 57.46 | 0.000 | 8.177748–8.755329 |
| _cons       |      | 7.589872 | .1632145 | 46.50  | 0.000 | 7.269977–7.909766 |
| peerrel2    | Peer | 1.109234 | .0975279 | 11.37  | 0.000 | .9180833–1.300385 |
| _cons       |      | 7.056996 | .1964299 | 35.93  | 0.000 | 6.672001–7.441992 |
| peerrel3    | Peer | 1.409361 | .1138314 | 12.38  | 0.000 | 1.186256–1.632467 |
| _cons       |      | 7.056996 | .1964299 | 35.93  | 0.000 | 6.672001–7.441992 |
| peerrel4    | Peer | 1.195982 | .0980272 | 12.20  | 0.000 | 1.003852–1.388112 |
| _cons       |      | 7.89358 | .169158  | 46.66  | 0.000 | 7.562036–8.225123 |
| parrel1     | Par  | 1 (constrained) | 9.368654 | .0819489 | 114.32 | 0.000 | 9.208037–9.529271 |
| _cons       |      | 9.287629 | .0903296 | 102.82 | 0.000 | 9.110587–9.464672 |
| parrel2     | Par  | 1.104355 | .1369365 | 8.06   | 0.000 | .8359649–1.372746 |
| _cons       |      | 9.287629 | .0903296 | 102.82 | 0.000 | 9.110587–9.464672 |
| parrel3     | Par  | 2.05859 | .2060583 | 9.99   | 0.000 | 1.654723–2.462457 |
| _cons       |      | 8.741898 | .136612 | 63.99  | 0.000 | 8.474144–9.009653 |
| parrel4     | Par  | 1.526706 | .1552486 | 9.83   | 0.000 | 1.222424–1.830987 |
| _cons       |      | 9.096609 | .1061607 | 85.69  | 0.000 | 8.888538–9.30468 |
| mean(Peer)  |      | .3296841 | .1570203 | 2.10   | 0.036 | .02193–.6374382 |
| mean(Par)   |      | -.0512439 | .0812255 | -0.63  | 0.531 | -.211619–.1091313 |
| var(e.peer-1) |    | 1.773813 | .1889104 | 9.21   | 0.000 | 1.439644–2.185549 |
| var(e.peer-2) |    | 2.165228 | .2321565 | 9.36   | 0.000 | 1.75484–2.671589 |
| var(e.peer-3) |    | 1.950679 | .2586196 | 7.58   | 0.000 | 1.504298–2.529516 |
| var(e.peer-4) |    | 1.822448 | .2151827 | 8.46   | 0.000 | 1.445942–2.296992 |
| var(e.parr-1) |   | 1.213159 | .192634 | 6.34   | 0.000 | 1.000547–1.470949 |
| var(e.parr-2) |   | 1.79031 | .1747374 | 9.75   | 0.000 | 1.478596–2.167739 |
| var(e.parr-3) |   | 1.015707 | .1713759 | 5.93   | 0.000 | .7297073–1.4138 |
| var(e.parr-4) |   | .8599648 | .1165865 | 7.36   | 0.000 | .6592987–1.121706 |
| var(Peer)   |       | 1.307976 | .206158 | 6.34   | 0.000 | .9603661–1.781406 |
| var(Par)    |       | .5201696 | .1029353 | 5.05   | 0.000 | .3529413–.766329 |
| cov(Peer,Par)|   | .3867156 | .079455 | 4.87   | 0.000 | .2309867–.5424445 |

**Notes:**

1. In *Which parameters vary by default, and which do not* in [SEM Intro 6], we wrote that, generally speaking, when we specify `group(groupvar)`, the measurement part of the model is constrained by default to be the same across the groups, whereas the remaining parts will have separate parameters for each group.
More precisely, we revealed that `sem` classifies each parameter into one of nine classes, which are the following:

<table>
<thead>
<tr>
<th>Class description</th>
<th>Class name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. structural coefficients</td>
<td>scoef</td>
</tr>
<tr>
<td>2. structural intercepts</td>
<td>scons</td>
</tr>
<tr>
<td>3. measurement coefficients</td>
<td>mcoef</td>
</tr>
<tr>
<td>4. measurement intercepts</td>
<td>mcons</td>
</tr>
<tr>
<td>5. covariances of structural errors</td>
<td>serrvar</td>
</tr>
<tr>
<td>6. covariances of measurement errors</td>
<td>merrvar</td>
</tr>
<tr>
<td>7. covariances between structural and measurement errors</td>
<td>smerrcov</td>
</tr>
</tbody>
</table>
| 8. means of exogenous variables     | meanex (*|)
| 9. covariances of exogenous variables| covex (*|)
| 10. all the above                   | all (*)    |
| 11. none of the above               | none       |

(*) Be aware that classes 8, 9, and 10 (meanex, covex, and all) exclude the observed exogenous variables—include only the latent exogenous variables—unless you specify option noxconditional or the noxconditional option is otherwise implied; see [SEM] `sem option noxconditional`. This is what you would desire in most cases.

By default, classes 3 and 4 are constrained to be equal and the rest are allowed to vary.

2. Thus you might expect that most of the parameters of our model would have been left unconstrained until you remember that we are fitting a measurement model. That is why `sem` listed 20 constraints at the top of the estimation results. Some of the constraints are substantive and some are normalization.

3. In the output, we have a separate table of parameter estimates for each level of `grade`.

   In our data, group 1 corresponds with students in grade 4, and group 2 corresponds with students in grade 5.

4. It may surprise you that the output contains estimates for the means of the latent variables. Usually, `sem` does not report this.

   Usually, you are running on only one group of data and those means cannot be estimated, at least not without additional identifying constraints. When you are running on two or more groups, the means for all the groups except one can be estimated.

In [SEM] `Example 21`, we use `estat ggof` to evaluate goodness of fit group by group.

In [SEM] `Example 22`, we use `estat ginvariant` to test whether parameters that are constrained across groups should not be and whether parameters that are not constrained could be.

In [SEM] `Example 23`, we show how to constrain the parameters we choose to be equal across groups.
Fitting the model with the Builder

Use the diagram above for reference.

1. Open the dataset.

   In the Command window, type
   
   . use https://www.stata-press.com/data/r16/sem_2fmmby

2. Open a new Builder diagram.

   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Create the measurement component for relationships with peers.

   Select the Add measurement component tool, \( \text{x} \), and then click in the diagram about halfway down from the top and about one-third of the way in from the left.

   In the resulting dialog box,
   
   a. change the Latent variable name to Peer;
   b. select peerrel1, peerrel2, peerrel3, and peerrel4 by using the Measurement variables control;
   c. select Left in the Measurement direction control;
   d. click on OK.

   If you wish, move the component by clicking on any variable and dragging it.

4. Create the measurement component for relationships with parents.

   Select the Add measurement component tool, \( \text{x} \), and then click in the diagram about halfway down from the top and about one-third of the way in from the right.

   In the resulting dialog box,
   
   a. change the Latent variable name to Par;
   b. select parrel1, parrel2, parrel3, and parrel4 by using the Measurement variables control;
   c. select Right in the Measurement direction control;
   d. click on OK.

   If you wish, move the component by clicking on any variable and dragging it.

5. Correlate the latent variables.

   a. Select the Add covariance tool, \( \text{\checkmark} \).
   
   b. Click in the upper-right quadrant of the Peer oval (it will highlight when you hover over it), and drag a covariance to the upper-left quadrant of the Par oval (it will highlight when you can release to connect the covariance).
6. Clean up.

If you do not like where a covariance has been connected to its variable, use the Select tool, , to click on the covariance, and then simply click on where it connects to an oval and drag the endpoint. You can also change the bow of the covariance by dragging the control point that extends from one end of the selected covariance.

7. Estimate.

Click on the Estimate button, , in the Standard Toolbar.

In the resulting dialog box, do the following:

a. Select the Group tab.

b. Select the Group analysis radio button. The variable grade should appear in the Group variable control.

c. Click on OK.

d. In the Standard Toolbar, use the Group control to toggle between results for group 1 and group 2.

You can open a completed diagram in the Builder by typing

   . webgetsem sem_2fmmby

Reference


Also see

[SEM] Example 3 — Two-factor measurement model
[SEM] Example 19 — Creating multiple-group summary statistics data
[SEM] Example 21 — Group-level goodness of fit
[SEM] Example 22 — Testing parameter equality across groups
[SEM] Example 23 — Specifying parameter constraints across groups
[SEM] Intro 6 — Comparing groups
[SEM] sem — Structural equation model estimation command
[SEM] sem group options — Fitting models on different groups
Example 21 — Group-level goodness of fit

Description

Below we demonstrate the `estat ggof` command, which may be used after `sem` with the `group()` option. `estat ggof` displays group-by-group goodness-of-fit statistics.

We pick up where [SEM] Example 20 left off:

```
use https://www.stata-press.com/data/r16/sem_2fmmby
sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4) (Par -> parrel1 parrel2 parrel3 parrel4), group(grade)
```

Remarks and examples

```
estat ggof
```

Group-level fit statistics

<table>
<thead>
<tr>
<th>grade</th>
<th>N</th>
<th>SRMR</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>134</td>
<td>0.063</td>
<td>0.969</td>
</tr>
<tr>
<td>2</td>
<td>251</td>
<td>0.047</td>
<td>0.955</td>
</tr>
</tbody>
</table>

Note: Group-level chi-squared statistics are not reported because of constraints between groups.

Notes:

1. Reported are the goodness-of-fit tests that `estat gof, stats(residuals)` would report. The difference is that they are reported for each group rather than overall.

2. If the fit is good, then SRMR (standardized root mean squared residual) will be close to 0 and CD (the coefficient of determination) will be near 1.

It is also appropriate to run `estat gof` to obtain overall results:

```
estat gof, stats(residuals)
```

<table>
<thead>
<tr>
<th>Fit statistic</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of residuals</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRMR</td>
<td>0.056</td>
<td>Standardized root mean squared residual</td>
</tr>
<tr>
<td>CD</td>
<td>0.958</td>
<td>Coefficient of determination</td>
</tr>
</tbody>
</table>

Also see

[SEM] Example 20 — Two-factor measurement model by group

[SEM] Example 4 — Goodness-of-fit statistics

[SEM] `estat ggof` — Group-level goodness-of-fit statistics

[SEM] `estat gof` — Goodness-of-fit statistics
Example 22 — Testing parameter equality across groups

Description

Below we demonstrate `estat ginvariant` to test parameters across groups.

We pick up where [SEM] Example 20 left off:

```
. use https://www.stata-press.com/data/r16/sem_2fmmby
. sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4) ///
     (Par -> parrel1 parrel2 parrel3 parrel4), group(grade)
```

Remarks and examples

We use `estat ginvariant` to test whether parameters that are constrained to be equal across groups should not be and whether parameters that are not constrained across groups could be.
Example 22 — Testing parameter equality across groups

Tests for group invariance of parameters

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Wald Test</th>
<th>Score Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>chi2</td>
<td>df</td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>peerrel1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>peerrel2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>peerrel3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>peerrel4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>parrel1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Par</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>parrel2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Par</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>parrel3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Par</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>parrel4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Par</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>_cons</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>var(e.peer</td>
<td>1)</td>
<td>0.024</td>
</tr>
<tr>
<td>var(e.peer</td>
<td>2)</td>
<td>0.033</td>
</tr>
<tr>
<td>var(e.peer</td>
<td>3)</td>
<td>0.011</td>
</tr>
<tr>
<td>var(e.peer</td>
<td>4)</td>
<td>0.294</td>
</tr>
<tr>
<td>var(e.parr</td>
<td>1)</td>
<td>1.981</td>
</tr>
<tr>
<td>var(e.parr</td>
<td>2)</td>
<td>14.190</td>
</tr>
<tr>
<td>var(e.parr</td>
<td>3)</td>
<td>0.574</td>
</tr>
<tr>
<td>var(e.parr</td>
<td>4)</td>
<td>0.022</td>
</tr>
<tr>
<td>var(Peer)</td>
<td>4.583</td>
<td>1</td>
</tr>
<tr>
<td>var(Par)</td>
<td>0.609</td>
<td>1</td>
</tr>
<tr>
<td>cov(Peer,Par)</td>
<td>0.780</td>
<td>1</td>
</tr>
</tbody>
</table>

Notes:

1. In the output above, score tests are reported for parameters that were constrained. The null hypothesis is that the constraint is valid. None of the tests reject a valid constraint.

2. Wald tests are reported for parameters that were not constrained. The null hypothesis is that a constraint would be valid. Only in two cases does it appear that grade 4 differs from grade 5, namely, the variance of e.parrel2 and the variance of Peer.
3. We remind you that these tests are marginal tests. That is, each test is intended to be interpreted separately. These are not joint tests of simultaneous imposition or relaxation of constraints. If you want simultaneous tests, you must do them yourself by using, for instance, the `test` command.

If joint tests of parameter classes are desired, the `class` option can be used.

These results imply that none of the constraints we impose should be relaxed, and that perhaps we could constrain all the variances and covariances to be equal across groups except for the variances of `e.parre12` and `Peer`. We do that in [SEM] Example 23.

Also see

[SEM] Example 20 — Two-factor measurement model by group
[SEM] Example 23 — Specifying parameter constraints across groups
[SEM] `estat ginvariant` — Tests for invariance of parameters across groups
Example 23 — Specifying parameter constraints across groups

Description

Below we demonstrate how to constrain the parameters we want constrained to be equal across groups when using `sem` with the `group()` option.

We pick up where [SEM] Example 22 left off:

```
. use https://www.stata-press.com/data/r16/sem_2fmmby
. sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4) ///
   (Par -> parrel1 parrel2 parrel3 parrel4), group(grade)
. estat ginvariant
```

The `estat ginvariant` command implied that perhaps we could constrain all the variances and covariances to be equal across groups except for the variances of `e.parrel2` and `Peer`.

Remarks and examples

Remarks are presented under the following headings:

   Background
   Fitting the constrained model

Background

We can specify which parameters we wish to allow to vary. Remember that `sem`’s `group()` option classifies the parameters of the model as follows:

<table>
<thead>
<tr>
<th>Class description</th>
<th>Class name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. structural coefficients</td>
<td>scoef</td>
</tr>
<tr>
<td>2. structural intercepts</td>
<td>scons</td>
</tr>
<tr>
<td>3. measurement coefficients</td>
<td>mcoef</td>
</tr>
<tr>
<td>4. measurement intercepts</td>
<td>mcons</td>
</tr>
<tr>
<td>5. covariances of structural errors</td>
<td>serrvar</td>
</tr>
<tr>
<td>6. covariances of measurement errors</td>
<td>merrvar</td>
</tr>
<tr>
<td>7. covariances between structural and measurement errors</td>
<td>smerrcov</td>
</tr>
</tbody>
</table>
| 8. means of exogenous variables            | meanex     (*)
| 9. covariances of exogenous variables      | covex      (*)
| 10. all the above                          | all        (*)
| 11. none of the above                      | none       (*)

(*) Exogenous variables means just the latent exogenous variables unless you specify `sem option noxconditional` or you specify option `method(mlmv)` (which implies option `noxconditional`); see [SEM] `sem option noxconditional`.

291
When fitting a model with the `group()` option,

```
   . sem ..., ... group(varname)
```

you may also specify the `ginvariant()` option:

```
   . sem ..., ... group(varname) ginvariant(class names)
```

You may specify any of the class names as being `ginvariant()`. You may specify as many class names as you wish. When you specify `ginvariant()`, `sem` cancels its default actions on which parameters vary and which do not, and uses the information you specify. All classes that you do not mention as being `ginvariant()` are allowed to vary across groups.

By using `ginvariant()`, you can constrain, or free by your silence, whole classes of parameters. For instance, you could type

```
   . sem ..., group(mygroup) ginvariant(mcoef mcons serrvar)
```

and you are constraining those parameters to be equal across groups and leaving unconstrained `scoef`, `scons`, `merrvar`, `smerrcov`, `meanex`, and `covex`.

In addition, if a class is constrained, you can still unconstrain individual coefficients. Consider the model

```
   . sem ... (x1<-L) ...
```

If you typed

```
   . sem ... (1: x1<-L@a1) (2: x1<-L@a2) ..., group(mygroup) ginvariant(all)
```

then all estimated parameters would be the same across groups except for the path `x1<-L`, and it would be free to vary in groups 1 and 2.

By the same token, if a class is unconstrained, you can still constrain individual coefficients. If you typed

```
   . sem ... (1: x1<-L@a) (2: x1<-L@a) ..., group(mygroup) ginvariant(none)
```

then you would leave unconstrained all parameters except the path `x1<-L`, and it would be constrained to be equal in groups 1 and 2.

This is all discussed in [SEM] Intro 6, including how to constrain and free variance and covariance parameters.

**Fitting the constrained model**

In our case, we wish to fit our model:

```
   . sem (Peer -> peerrel1 peerrel2 peerrel3 peerrel4) ///
   (Par -> parrel1 parrel2 parrel3 parrel4), ///
   group(grade)
```
We impose constraints on all parameters except the variances of $e.parrel2$ and $Peer$. We can do that by typing

```
.se (Peer -> peerrel1 peerrel2 peerrel3 peerrel4)
> (Par -> parrel1 parrel2 parrel3 parrel4),
> group(grade)
> ginvariant(all)
> byparm
> var(1: e.parrel2@v1)
> var(2: e.parrel2@v2)
> var(1: Peer@v3)
> var(2: Peer@v4)
```

Endogenous variables

Measurement: peerrel1 peerrel2 peerrel3 peerrel4 parrel1 parrel2 parrel3 parrel4

Exogenous variables

Latent: Peer Par

Fitting target model:

Iteration 0: log likelihood = -5560.9934
Iteration 1: log likelihood = -5552.3122
Iteration 2: log likelihood = -5549.5391
Iteration 3: log likelihood = -5549.3528
Iteration 4: log likelihood = -5549.3501
Iteration 5: log likelihood = -5549.3501

Structural equation model

Number of obs = 385
Grouping variable = grade
Number of groups = 2
Estimation method = ml
Log likelihood = -5549.3501

( 1) [peerrel1]1bn.grade#c.Peer = 1
( 2) [peerrel2]1bn.grade#c.Peer - [peerrel2]2.grade#c.Peer = 0
( 3) [peerrel3]1bn.grade#c.Peer - [peerrel3]2.grade#c.Peer = 0
( 4) [peerrel4]1bn.grade#c.Peer - [peerrel4]2.grade#c.Peer = 0
( 5) [parrel1]1bn.grade#c.Par = 1
( 6) [parrel2]1bn.grade#c.Par - [parrel2]2.grade#c.Par = 0
( 7) [parrel3]1bn.grade#c.Par - [parrel3]2.grade#c.Par = 0
( 8) [parrel4]1bn.grade#c.Par - [parrel4]2.grade#c.Par = 0
( 9) [/]var(e.peerrel1)#1bn.grade - [/]var(e.peerrel1)#2.grade = 0
(10) [/]var(e.peerrel2)#1bn.grade - [/]var(e.peerrel2)#2.grade = 0
(11) [/]var(e.peerrel3)#1bn.grade - [/]var(e.peerrel3)#2.grade = 0
(12) [/]var(e.peerrel4)#1bn.grade - [/]var(e.peerrel4)#2.grade = 0
(13) [/]var(e.parrel1)#1bn.grade - [/]var(e.parrel1)#2.grade = 0
(14) [/]var(e.parrel3)#1bn.grade - [/]var(e.parrel3)#2.grade = 0
(15) [/]var(e.parrel4)#1bn.grade - [/]var(e.parrel4)#2.grade = 0
(16) [/]cov(Peer,Par)#1bn.grade - [/]cov(Peer,Par)#2.grade = 0
(17) [/]var(Peer)#1bn.grade - [/]var(Peer)#2.grade = 0
(18) [peerrel1]1bn.grade - [peerrel1]2.grade = 0
(19) [peerrel2]1bn.grade - [peerrel2]2.grade = 0
(20) [peerrel3]1bn.grade - [peerrel3]2.grade = 0
(21) [peerrel4]1bn.grade - [peerrel4]2.grade = 0
(22) [parrel1]1bn.grade - [parrel1]2.grade = 0
(23) [parrel2]1bn.grade - [parrel2]2.grade = 0
(24) [parrel3]1bn.grade - [parrel3]2.grade = 0
(25) [parrel4]1bn.grade - [parrel4]2.grade = 0
(26) [peerrel1]2.grade#c.Peer = 1
(27) [parrel1]2.grade#c.Par = 1
| Parameter   | Coef.   | Std. Err. | z      | P>|z|   | [95% Conf. Interval] |
|------------|---------|-----------|--------|-------|----------------------|
| **Measurement** |         |           |        |       |                      |
| peerrel1   |         |           |        |       |                      |
| Peer      | [*]    | 1 (constrained) |      |       |                      |
| _cons     | [*]    | 8.708274  | .0935844 | 93.05 | 0.000 | 8.524852 8.891696 |
| peerrel2   |         |           |        |       |                      |
| Peer      | [*]    | 1.112225  | .0973506 | 11.42 | 0.000 | 1.021421 1.203029 |
| _cons     | [*]    | 7.858713  | .1035989 | 75.86 | 0.000 | 7.655663 8.061763 |
| peerrel3   |         |           |        |       |                      |
| Peer      | [*]    | 1.416486  | .113489  | 12.48 | 0.000 | 1.194052 1.638921 |
| _cons     | [*]    | 7.398217  | .1147474 | 64.47 | 0.000 | 7.173316 7.623118 |
| peerrel4   |         |           |        |       |                      |
| Peer      | [*]    | 1.196494  | .0976052 | 12.26 | 0.000 | 1.005191 1.387796 |
| _cons     | [*]    | 8.183148  | .1021513 | 80.11 | 0.000 | 7.982936 8.383361 |
| **Par**    |         |           |        |       |                      |
| parrel1    |         |           |        |       |                      |
| Par       | [*]    | 1 (constrained) |      |       |                      |
| _cons     | [*]    | 9.339558  | .0648742 | 143.96| 0.000 | 9.212407 9.46671  |
| parrel2    |         |           |        |       |                      |
| Par       | [*]    | 1.100315  | .1362999 | 8.07  | 0.000 | .833172 1.367458  |
| _cons     | [*]    | 9.255299  | .0725417 | 127.59| 0.000 | 9.11312 9.397478 |
| parrel3    |         |           |        |       |                      |
| Par       | [*]    | 2.051278  | .2066714 | 9.93  | 0.000 | 1.64621 2.456347 |
| _cons     | [*]    | 8.676961  | .088927  | 97.57 | 0.000 | 8.502667 8.851255 |
| parrel4    |         |           |        |       |                      |
| Par       | [*]    | 1.529938  | .154971  | 9.87  | 0.000 | 1.2262 1.833675  |
| _cons     | [*]    | 9.045247  | .0722358 | 125.22| 0.000 | 8.903667 9.186826 |
Example 23 — Specifying parameter constraints across groups

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate 1</th>
<th>Estimate 2</th>
<th>Estimate 3</th>
<th>Estimate 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(e.peer~1)</td>
<td>1.799133</td>
<td>.159059</td>
<td>1.512898</td>
<td>2.139523</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.peer~2)</td>
<td>2.186953</td>
<td>.193911</td>
<td>1.838086</td>
<td>2.602035</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.peer~3)</td>
<td>1.915661</td>
<td>.2129913</td>
<td>1.54056</td>
<td>2.382094</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.peer~4)</td>
<td>1.767354</td>
<td>.1746104</td>
<td>1.45622</td>
<td>2.144965</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.parr~1)</td>
<td>1.125082</td>
<td>.0901338</td>
<td>.9615942</td>
<td>1.316366</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.parr~2)</td>
<td>.9603043</td>
<td>.13383</td>
<td>.730775</td>
<td>1.261927</td>
</tr>
<tr>
<td>1</td>
<td>1.951555</td>
<td>.3387796</td>
<td>1.388727</td>
<td>2.742489</td>
</tr>
<tr>
<td>2</td>
<td>1.361431</td>
<td>.2122853</td>
<td>1.002927</td>
<td>1.848084</td>
</tr>
<tr>
<td>var(e.parr~3)</td>
<td>.9606889</td>
<td>.1420406</td>
<td>.7190021</td>
<td>1.283617</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.parr~4)</td>
<td>.8496935</td>
<td>.0933448</td>
<td>.6850966</td>
<td>1.053835</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(Peer)</td>
<td>1.961555</td>
<td>.3387796</td>
<td>1.388727</td>
<td>2.742489</td>
</tr>
<tr>
<td>1</td>
<td>1.951555</td>
<td>.3387796</td>
<td>1.388727</td>
<td>2.742489</td>
</tr>
<tr>
<td>2</td>
<td>1.361431</td>
<td>.2122853</td>
<td>1.002927</td>
<td>1.848084</td>
</tr>
<tr>
<td>var(Par)</td>
<td>.4952527</td>
<td>.0927994</td>
<td>.3430288</td>
<td>.7150281</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cov(Peer,Par)</td>
<td>.4096197</td>
<td>.0708726</td>
<td>5.78</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>[*]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: [*] identifies parameter estimates constrained to be equal across groups.

LR test of model vs. saturated: chi2(61) = 75.25, Prob > chi2 = 0.1037

Notes:

1. In [SEM] Example 20, we previously fit this model by typing
   
   . sem (...) (...) , group(grade)

   This time, we typed
   
   . sem (...) (...) , group(grade) ///
   ginvariant(all) ///
   bypass ///
   var(1: e.parrel2@v1) ///
   var(2: e.parrel2@v2) ///
   var(1: Peer@v3) ///
   var(2: Peer@v4)

2. We specified the bypass option so that the results are sorted by parameters rather than groups. With this option, sem will report a single value for a parameter estimate if it is constrained to be equal across the groups. Because most of the parameters are constrained to be equal across groups, the output is much shorter than the default output. The default output produces a separate table for each group.

3. Previously, sem, group() mentioned 20 constraints that it imposed because of normalization or because of assumed ginvariant(mcoef mcons).

   This time, sem, group() mentioned 27 constraints. It applied more constraints because we specified ginvariant(all).
4. After the `ginvariant(all)` option, we relaxed the following constraints:

\[
\begin{align*}
\text{var}(1: &e.\text{parrel2}@v1) \\
\text{var}(2: &e.\text{parrel2}@v2) \\
\text{var}(1: &\text{Peer}@v3) \\
\text{var}(2: &\text{Peer}@v4)
\end{align*}
\]

`ginvariant(all)` specified, among other constraints, that

\[
\begin{align*}
\text{var}(1: &e.\text{parrel2}) == \text{var}(2: e.\text{parrel2}) \\
\text{var}(1: &\text{Peer}) == \text{var}(2: \text{Peer})
\end{align*}
\]

`ginvariant(all)` did that by secretly issuing the options

\[
\begin{align*}
\text{var}(1: &e.\text{parrel2}@\text{secretname1}) \\
\text{var}(2: &e.\text{parrel2}@\text{secretname1}) \\
\text{var}(1: &\text{Peer}@\text{secretname2}) \\
\text{var}(2: &\text{Peer}@\text{secretname2})
\end{align*}
\]

because that is how you impose equality constraints with the path notation. When we specified

\[
\begin{align*}
\text{var}(1: &e.\text{parrel2}@v1) \\
\text{var}(2: &e.\text{parrel2}@v2) \\
\text{var}(1: &\text{Peer}@v3) \\
\text{var}(2: &\text{Peer}@v4)
\end{align*}
\]

our new constraints overrode the secretly issued constraints. It would not have worked to leave off the symbolic names; see [SEM] *sem path notation extensions*. We specified the symbolic names \(v1, v2, v3, \text{ and } v4\). \(v1\) and \(v2\) overrode \text{secretname1}, and thus the constraint that \(\text{var}(e.\text{parrel2})\) be equal across the two groups was relaxed. \(v3\) and \(v4\) overrode \text{secretname2}, and thus the constraint that \(\text{var}(\text{Peer})\) be equal across groups was relaxed.

**Also see**

- [SEM] Example 20 — Two-factor measurement model by group
- [SEM] Example 22 — Testing parameter equality across groups
- [SEM] Intro 6 — Comparing groups
- [SEM] *sem group options* — Fitting models on different groups
Description

Below we demonstrate sem’s reliability() option with the following data:

```
. use https://www.stata-press.com/data/r16/sem_rel
    (measurement error with known reliabilities)
. summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1,234</td>
<td>701.081</td>
<td>71.79378</td>
<td>487</td>
<td>943</td>
</tr>
<tr>
<td>x1</td>
<td>1,234</td>
<td>100.278</td>
<td>14.1552</td>
<td>51</td>
<td>149</td>
</tr>
<tr>
<td>x2</td>
<td>1,234</td>
<td>100.2066</td>
<td>14.50912</td>
<td>55</td>
<td>150</td>
</tr>
</tbody>
</table>
```

. notes
   _dta:
   1. Fictional data.
   2. Variables x1 and x2 each contain a test score designed to measure X. The
test is scored to have mean 100.
   3. Variables x1 and x2 are both known to have reliability 0.5.
   4. Variable y is the outcome, believed to be related to X.

See [SEM] sem and gsem option reliability() for background.

Remarks and examples

Remarks are presented under the following headings:

- Baseline model (reliability ignored)
- Model with reliability
- Model with two measurement variables and reliability
Baseline model (reliability ignored)

```
. sem (y <- x1)
Endogenous variables
Observed:  y
Exogenous variables
Observed:  x1
Fitting target model:
Iteration 0:  log likelihood = -11629.745
Iteration 1:  log likelihood = -11629.745
Structural equation model  Number of obs  =  1,234
Estimation method = ml
Log likelihood = -11629.745

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x1</td>
<td>3.54976</td>
<td>.1031254</td>
<td>34.42</td>
<td>0.000</td>
<td>3.347637 3.751882</td>
</tr>
<tr>
<td>_cons</td>
<td>345.1184</td>
<td>10.44365</td>
<td>33.05</td>
<td>0.000</td>
<td>324.6492 365.5876</td>
</tr>
<tr>
<td>var(e.y)</td>
<td>2627.401</td>
<td>105.7752</td>
<td>2428.053</td>
<td>2843.115</td>
<td></td>
</tr>
</tbody>
</table>
```

LR test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .

Notes:
1. In these data, variable x1 is measured with error.
2. If we ignore that, we obtain a path coefficient for y<-x1 of 3.55.
3. We also ran this model for y<-x2. We obtained a path coefficient of 3.48.
Model with reliability

```
Example 24 — Reliability 299

. sem (x1<-X) (y<-X), reliability(x1 .5)
Endogenous variables
Measurement:  x1 y
Exogenous variables
Latent:  X
Fitting target model:
Iteration 0:  log likelihood = -11745.845
Iteration 1:  log likelihood = -11661.626
Iteration 2:  log likelihood = -11631.469
Iteration 3:  log likelihood = -11629.755
Iteration 4:  log likelihood = -11629.745
Iteration 5:  log likelihood = -11629.745

Structural equation model
Estimation method = ml
Log likelihood = -11629.745
( 1)  [x1]X = 1
( 2)  [\]var(e.x1) = 100.1036

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.  Std. Err.  z  P&gt;</td>
<td>z</td>
<td>[95% Conf. Interval]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_cons</td>
<td>100.278          .4027933    248.96       0.000       99.4885   101.0674</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>7.09952          .352463     20.14       0.000       6.408705   7.790335</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_cons</td>
<td>701.081          2.042929    343.17       0.000       697.077    705.0851</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.x1)</td>
<td>100.1036 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(e.y)</td>
<td>104.631           207.3381    4.1599      0.000       2.159334   5086.411</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(X)</td>
<td>100.1036          8.060038    8.5932      0.000       85.48963   117.2157</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

LR test of model vs. saturated:  chi2(0) = 0.00, Prob > chi2 = .
```

Notes:

1. We wish to estimate the effect of y<-x1 when x1 is measured with error (0.50 reliability). To do that, we introduce latent variable X and write our model as (x1<-X) (y<-X).

2. When we ignored the measurement error of x1, we obtained a path coefficient for y<-x1 of 3.55. Taking into account the measurement error, we obtain a coefficient of 7.1.
Model with two measurement variables and reliability

```
. sem (x1 x2<-X) (y<-X), reliability(x1 .5 x2 .5)
Endogenous variables
Measurement: x1 x2 y
Exogenous variables
Latent: X
Fitting target model:
Iteration 0: log likelihood = -16258.636
Iteration 1: log likelihood = -16258.401
Iteration 2: log likelihood = -16258.4
Structural equation model Number of obs = 1,234
Estimation method = ml
Log likelihood = -16258.4
( 1) [x1]X = 1
( 2) [\]var(e.x1) = 100.1036
( 3) [\]var(e.x2) = 105.1719

                              OIM
                      Coef.  Std. Err.     z  P>|z|  [95% Conf. Interval]
Measurement
  x1
    X       1  (constrained)
    _cons   100.278    .4037851  248.34    0.000   99.48655   101.0694
  x2
    X       1.030101    .0417346  24.68    0.000    .9483029   1.1119
    _cons   100.2066    .4149165  241.51    0.000   99.39342  101.0199
  y
    X        7.031299    .2484176  28.30    0.000    6.544409   7.518188
    _cons   701.0812    2.042928  343.17    0.000   697.077    705.0851
  var(e.x1)  100.1036 (constrained)
  var(e.x2)  105.1719 (constrained)
  var(e.y)   152.3290  105.2600    39.31868 590.1553
  var(X)    101.0907  7.343656  87.67509 116.5591

LR test of model vs. saturated: chi2(2) = 0.59, Prob > chi2 = 0.7430
```

Notes:

1. We wish to estimate the effect of y<-X. We have two measures of X—x1 and x2—both measured with error (0.50 reliability).
2. In the previous section, we used just x1. We obtained path coefficient 7.1 with standard error 0.4. Using both x1 and x2, we obtain path coefficient 7.0 and standard error 0.2.
3. We at StataCorp created these fictional data. The true coefficient is 7.

Also see

[SEM] sem and gsem option reliability() — Fraction of variance not due to measurement error
[SEM] Example 1 — Single-factor measurement model
Example 25 — Creating summary statistics data from raw data

Description

Below we show how to create summary statistics data (SSD) from raw data. We use auto2.dta:

```
use https://www.stata-press.com/data/r16/auto2
(1978 Automobile Data)
describe
(output omitted)
summarize
(output omitted)
```

Remarks and examples

Remarks are presented under the following headings:

- Preparing data for conversion
- Converting to summary statistics form
- Publishing SSD
- Creating SSD with multiple groups

We are going to create SSD containing the variables `price`, `mpg`, `weight`, `displacement`, and `foreign`.

Preparing data for conversion

Before building the SSD, prepare the data to be converted:

1. Drop variables that you do not intend to include in the SSD. Dropping variables is not a requirement, but it will be easier to spot problems if you begin by eliminating the irrelevant variables.
2. Verify that you have no string variables in the resulting data. Summary statistics datasets cannot contain string values.
3. Verify that there are no missing values. If there are, be aware that observations containing one or more variables with missing values will be omitted from the SSD.
4. Verify that all variables are on a reasonable scale. We recommend that the means of variables be only 3 or 4 orders of magnitude different from each other. This will help to preserve numerical accuracy when the SSD are used.
5. Create any new variables containing transformations of existing variables that might be useful later. Once the data are converted to summary statistics form, you will not be able to create such variables.
6. Place the variables in a logical order. That will help the user of the SSD understand the data.
7. Save the resulting prepared data. Probably you will never need the prepared data, but one never knows for sure.
We take our own advice below:

* Suggestion 1: Keep relevant variables:
* keep price mpg weight displacement foreign

* Suggestion 2: Check for string variables
* Suggestion 3: Verify no missing values
* Suggestion 4: Verify variables on a reasonable scale:
* summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>74</td>
<td>6165.257</td>
<td>2949.496</td>
<td>3291</td>
<td>15906</td>
</tr>
<tr>
<td>mpg</td>
<td>74</td>
<td>21.2973</td>
<td>5.785503</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>weight</td>
<td>74</td>
<td>3019.459</td>
<td>777.1936</td>
<td>1760</td>
<td>4840</td>
</tr>
<tr>
<td>displacement</td>
<td>74</td>
<td>197.2973</td>
<td>91.83722</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>74</td>
<td>0.2972973</td>
<td>0.4601885</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

* We will rescale weight and price:
* replace weight = weight/1000
variable weight was int now float
(74 real changes made)
* replace price = price/1000
variable price was int now float
(74 real changes made)
* label var weight "Weight (1000s lbs.)"
* label var price "Price ($1,000s)"
* and now we check our work:
* summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>74</td>
<td>6.165257</td>
<td>2.949496</td>
<td>3.291</td>
<td>15.906</td>
</tr>
<tr>
<td>mpg</td>
<td>74</td>
<td>21.2973</td>
<td>5.785503</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>weight</td>
<td>74</td>
<td>3.019459</td>
<td>.7771936</td>
<td>1.76</td>
<td>4.84</td>
</tr>
<tr>
<td>displacement</td>
<td>74</td>
<td>197.2973</td>
<td>91.83722</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>74</td>
<td>0.2972973</td>
<td>0.4601885</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

* Suggestion 5: Create useful transformations:
* generate gpm = 1/mpg
* label var gpm "Gallons per mile"
Example 25 — Creating summary statistics data from raw data

* Suggestion 6: Place variables in logical order:

. order price mpg gpm
. summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>74</td>
<td>6.165257</td>
<td>2.949496</td>
<td>3.291</td>
<td>15.906</td>
</tr>
<tr>
<td>mpg</td>
<td>74</td>
<td>21.2973</td>
<td>5.785503</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>gpm</td>
<td>74</td>
<td>0.0501928</td>
<td>0.0127986</td>
<td>0.0243902</td>
<td>0.0833333</td>
</tr>
<tr>
<td>weight</td>
<td>74</td>
<td>3.019459</td>
<td>0.7771936</td>
<td>1.76</td>
<td>4.84</td>
</tr>
<tr>
<td>displacement</td>
<td>74</td>
<td>197.2973</td>
<td>91.83722</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>74</td>
<td>0.2972973</td>
<td>0.4601885</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

* Suggestion 7: save prepared data

. save auto_raw
file auto_raw.dta saved

Converting to summary statistics form

To create the summary statistics dataset, you just need to type `ssd build` and the names of the variables to be included. If you have previously kept the relevant variables, you can type `ssd build _all`.

We recommend the following steps:

1. Convert data to summary statistics form:
   . ssd build _all

2. Review the result:
   . ssd describe
   . notes
   . ssd list

3. Digitally sign the data:
   . datasignature set

4. Save the data:
   . save auto_ss

We follow our advice below. After that, we will show you the advantages of digitally signing the data.

. * Convert data:
. * ssd build _all
(data in memory now summary statistics data; you can use `ssd describe` and `ssd list` to describe and list results.)
Example 25 — Creating summary statistics data from raw data

* Review results:
* 
* ssd describe
Summary statistics data
obs: 74
vars: 6
(_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>Price ($1,000s)</td>
</tr>
<tr>
<td>mpg</td>
<td>Mileage (mpg)</td>
</tr>
<tr>
<td>gpm</td>
<td>Gallons per mile</td>
</tr>
<tr>
<td>weight</td>
<td>Weight (1000s lbs.)</td>
</tr>
<tr>
<td>displacement</td>
<td>Displacement (cu. in.)</td>
</tr>
<tr>
<td>foreign</td>
<td>Car type</td>
</tr>
</tbody>
</table>

* notes
_dta:
1. summary statistics data built from 'auto_raw.dta' on 25 Feb 2019 17:30:30 using -ssd build _all-

* ssd list
Observations = 74
Means:

<table>
<thead>
<tr>
<th></th>
<th>price</th>
<th>mpg</th>
<th>gpm</th>
<th>weight</th>
<th>displacement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.165257</td>
<td>21.297297</td>
<td>.0501928</td>
<td>3.0194595</td>
<td>197.2973</td>
</tr>
</tbody>
</table>

foreign .2972973

Variances implicitly defined; they are the diagonal of the covariance matrix.

Covariances:

<table>
<thead>
<tr>
<th></th>
<th>price</th>
<th>mpg</th>
<th>gpm</th>
<th>weight</th>
<th>displacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>8.6995258</td>
<td>33.472047</td>
<td>.0001638</td>
<td>.06402985</td>
<td>134.06705</td>
</tr>
<tr>
<td>mpg</td>
<td>-7.9962828</td>
<td>-.6294262</td>
<td>.00849897</td>
<td>.6387345</td>
<td>8434.0748</td>
</tr>
<tr>
<td>gpm</td>
<td>.02178417</td>
<td>-.06991586</td>
<td>.0001638</td>
<td>.60402985</td>
<td>25.938912</td>
</tr>
<tr>
<td>weight</td>
<td>1.2346748</td>
<td>1.0473899</td>
<td>-.0212897</td>
<td>-.21202888</td>
<td>.60402985</td>
</tr>
<tr>
<td>displacement</td>
<td>.21177342</td>
<td>.21177342</td>
<td>.21177342</td>
<td>.21177342</td>
<td>.21177342</td>
</tr>
</tbody>
</table>

* Digitally sign:
* 
* datasignature set
8:8(102846):1914186416:2867097560 (data signature set)

* Save:
* 
* save auto_ss
file auto_ss.dta saved

We recommend digitally signing the data. This way, anyone can verify later that the data are unchanged:

* datasignature confirm
(data unchanged since 29may2018 15:18)
Let us show you what would happen if the data had changed:

```
. replace mpg = mpg+.0001 in 5
   (1 real change made)
. datasignature confirm
   data have changed since 29may2018 15:18
   r(9);
```

There is no reason for you or anyone else to change the SSD after it has been created, so we recommend that you digitally sign the data. With regular datasets, users do make changes, if only by adding variables.

Be aware that the data signature is a function of the variable names, so if you rename a variable—something you are allowed to do—the signature will change and `datasignature` will report, for example, “data have changed since 29may2018 15:18”. Solutions to that problem are discussed in [SEM] ssd.

**Publishing SSD**

The summary statistics dataset you have just created can obviously be sent to and used by any Stata user. If you wish to publish your data in printed form, use `ssd describe` and `ssd list` to describe and list the data.

**Creating SSD with multiple groups**

The process for creating SSD containing multiple groups is nearly the same as for creating single-group data. The only differences are that you do not drop the group variable during preparation and that rather than typing

```
. ssd build _all
```
you type

```
. ssd build _all, group(varname)
```

Below we build the automobile SSD again, but this time, we specify `group(rep78)`:

```
. ssd build _all, group(rep78)
```

If you think carefully about this, you may be worried that `_all` includes `rep78` and thus we will be including the grouping variable among the summary statistics. `ssd build` knows to omit the group variable:

```
* Suggestion 1: Keep relevant variables:
*  
* webuse auto2, clear
(1978 Automobile Data)
 . keep price mpg weight displacement foreign rep78
```
```
. * Suggestion 2: Check for string variables
. * Suggestion 3: Verify no missing values
. * Suggestion 4: Verify variables on a reasonable scale:
. *
. summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>74</td>
<td>6165.257</td>
<td>2949.496</td>
<td>3291</td>
<td>15906</td>
</tr>
<tr>
<td>mpg</td>
<td>74</td>
<td>21.2973</td>
<td>5.785503</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>rep78</td>
<td>69</td>
<td>3.405797</td>
<td>.9899323</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>weight</td>
<td>74</td>
<td>3019.459</td>
<td>777.1936</td>
<td>1760</td>
<td>4840</td>
</tr>
<tr>
<td>displacement</td>
<td></td>
<td>197.2973</td>
<td>91.83722</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>74</td>
<td>.2972973</td>
<td>.4601885</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

. drop if rep78 >= . (5 observations deleted)
. * We will rescale weight and price:
. replace weight = weight/1000
variable weight was int now float (69 real changes made)
. replace price = price/1000
variable price was int now float (69 real changes made)
. label var weight "Weight (1000s lbs.)"
. label var price "Price ($1,000s)"
. * and now we check our work:
. summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>69</td>
<td>6.146043</td>
<td>2.91244</td>
<td>3.291</td>
<td>15.906</td>
</tr>
<tr>
<td>mpg</td>
<td>69</td>
<td>21.28986</td>
<td>5.866408</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>rep78</td>
<td>69</td>
<td>3.405797</td>
<td>.9899323</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>weight</td>
<td>69</td>
<td>3.032029</td>
<td>.7928515</td>
<td>1.76</td>
<td>4.84</td>
</tr>
<tr>
<td>displacement</td>
<td></td>
<td>198</td>
<td>93.14789</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>69</td>
<td>.3043478</td>
<td>.4635016</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

. * Suggestion 5: Create useful transformations:
. *
. generate gpm = 1/mpg
. label var gpm "Gallons per mile"
. * Suggestion 6: Place variables in logical order:
. *
. order price mpg gpm
. summarize

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>69</td>
<td>6.146043</td>
<td>2.91244</td>
<td>3.291</td>
<td>15.906</td>
</tr>
<tr>
<td>mpg</td>
<td>69</td>
<td>21.28986</td>
<td>5.866408</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>gpm</td>
<td>69</td>
<td>.0502584</td>
<td>.0128353</td>
<td>.0243902</td>
<td>.0833333</td>
</tr>
<tr>
<td>rep78</td>
<td>69</td>
<td>3.405797</td>
<td>.9899323</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>weight</td>
<td>69</td>
<td>3.032029</td>
<td>.7928515</td>
<td>1.76</td>
<td>4.84</td>
</tr>
<tr>
<td>displacement</td>
<td></td>
<td>198</td>
<td>93.14789</td>
<td>79</td>
<td>425</td>
</tr>
<tr>
<td>foreign</td>
<td>69</td>
<td>.3043478</td>
<td>.4635016</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```
Example 25 — Creating summary statistics data from raw data

* Suggestion 7: save prepared data
.
* save auto_group_raw
file auto_group_raw.dta saved
.
* Convert data:
*
ssd build _all, group(rep78)
(data in memory now summary statistics data; you can use ssd describe and
ssd list to describe and list results.)
.
* Review results:
*
ssd describe

Summary statistics data
obs: 69
vars: 6

<table>
<thead>
<tr>
<th>variable name</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>Price ($1,000s)</td>
</tr>
<tr>
<td>mpg</td>
<td>Mileage (mpg)</td>
</tr>
<tr>
<td>gpm</td>
<td>Gallons per mile</td>
</tr>
<tr>
<td>weight</td>
<td>Weight (1000s lbs.)</td>
</tr>
<tr>
<td>displacement</td>
<td>Displacement (cu. in.)</td>
</tr>
<tr>
<td>foreign</td>
<td>Car type</td>
</tr>
</tbody>
</table>

Group variable: rep78 (5 groups)
Obs. by group: 2, 8, 30, 18, 11

notes
_.dta:
1. summary statistics data built from ‘auto_group_raw.dta’ on 25 Feb 2019
17:30:30 using -ssd build _all, group(rep78)-

ssd list

Group rep78==Poor:
(output omitted)

Group rep78==Fair:
(output omitted)

Group rep78==Average:
(output omitted)

Group rep78==Good:
(output omitted)

Group rep78==Excellent:
Observations = 11
Means:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>mpg</td>
<td>gpm</td>
<td>weight</td>
<td>displacement</td>
</tr>
<tr>
<td>5.913</td>
<td>27.3636</td>
<td>.040481</td>
<td>2.3227</td>
<td>111.0909</td>
</tr>
<tr>
<td>foreign</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.81818182</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Variances implicitly defined; they are the diagonal of the covariance matrix.

Covariances:

<table>
<thead>
<tr>
<th></th>
<th>price</th>
<th>mpg</th>
<th>gpm</th>
<th>weight</th>
<th>displacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>6.8422143</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpg</td>
<td>-15.608899</td>
<td>76.254545</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gpm</td>
<td>.02750797</td>
<td>-.1184875</td>
<td>.00019114</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>.956802</td>
<td>-3.0610912</td>
<td>.00510833</td>
<td>.16856184</td>
<td></td>
</tr>
<tr>
<td>displacement</td>
<td>55.493298</td>
<td>-201.03636</td>
<td>.33150758</td>
<td>9.9577283</td>
<td>648.09091</td>
</tr>
<tr>
<td>foreign</td>
<td>.34169998</td>
<td>-.92727273</td>
<td>.00182175</td>
<td>.07254547</td>
<td>3.8181818</td>
</tr>
<tr>
<td>.16363636</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Also see

[SEM] ssd — Making summary statistics data (sem only)
Example 26 — Fitting a model with data missing at random

Description

sem method(mlmv) is demonstrated using

```
  . use https://www.stata-press.com/data/r16/cfa_missing
  (CFA MAR data)
  . summarize
                  Variable |     Obs  Mean   Std. Dev.    Min   Max
                    id    500  250.5     144.4818     1   500
      test1       406  97.37475    13.91442   56.0406  136.5672
      test2       413  98.04501    13.84145   62.25496  129.3881
      test3       443 100.9699     13.4862   65.51753  137.3046
      test4       417  99.56815    13.84145   53.8719  153.9779
                  taken    500   3.358    0.6593219     2    4
```  

_dta:

1. Fictional data on 500 subjects taking four tests.
3. 230 took all 4 tests
4. 219 took 3 of the 4 tests
5. 51 took 2 of the 4 tests
6. All tests have expected mean 100, s.d. 14.

See [SEM] Intro 4 for background.

Remarks and examples

Remarks are presented under the following headings:

Fitting the model with method(ml)
Fitting the model with method(mlmv)
Fitting the model with the Builder
Fitting the model with method(ml)

We fit a single-factor measurement model.

```
. sem (test1 test2 test3 test4 <- X), nolog
(270 observations with missing values excluded)
Endogenous variables
Measurement:  test1 test2 test3 test4
Exogenous variables
Latent:  X
Structural equation model  Number of obs  =    230
Estimation method  =  ml
Log likelihood  =  -3464.3099
  ( 1)  [test1]X = 1

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.  Std. Err. z  P&gt;</td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>test1</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1 (constrained)</td>
</tr>
<tr>
<td>_cons</td>
<td>96.76907  .8134878  118.96  0.000  95.17467  98.36348</td>
</tr>
<tr>
<td>test2</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1.021885  .1183745  8.63  0.000  .789875  1.253895</td>
</tr>
<tr>
<td>_cons</td>
<td>92.41248  .8405189  109.95  0.000  90.7651  94.05987</td>
</tr>
<tr>
<td>test3</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>.5084673  .0814191  6.25  0.000  .3488889  .6680457</td>
</tr>
<tr>
<td>_cons</td>
<td>94.12958  .7039862  133.71  0.000  92.7498  95.50937</td>
</tr>
<tr>
<td>test4</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>.5585651  .0857772  6.51  0.000  .3904449  .7266853</td>
</tr>
<tr>
<td>_cons</td>
<td>92.2556  .7322511  125.99  0.000  90.82042  93.69079</td>
</tr>
<tr>
<td>var(e.test1)</td>
<td>55.86083  10.85681</td>
</tr>
<tr>
<td>var(e.test2)</td>
<td>61.88092  11.50377</td>
</tr>
<tr>
<td>var(e.test3)</td>
<td>89.07839  8.962574</td>
</tr>
<tr>
<td>var(e.test4)</td>
<td>93.26508  9.504276</td>
</tr>
<tr>
<td>var(X)</td>
<td>96.34453  16.28034</td>
</tr>
</tbody>
</table>

LR test of model vs. saturated: chi2(2)  =  0.39, Prob > chi2 = 0.8212
```
Notes:

1. This model was fit using 230 of the 500 observations in the dataset. Unless you use sem’s `method(mlmv)`, observations are casewise omitted, meaning that if there is a single variable with a missing value among the variables being used, the observation is ignored.

2. The coefficients for `test3` and `test4` are 0.51 and 0.56. Because we at StataCorp manufactured these data, we can tell you that the true coefficients are 1.

3. The error variance for `e.test1` and `e.test2` are understated. These data were manufactured with an error variance of 100.

4. These data are missing at random (MAR), not missing completely at random (MCAR). In MAR data, which values are missing can be a function of the observed values in the data. MAR data can produce biased estimates if the missingness is ignored, as we just did. MCAR data do not bias estimates.

Fitting the model with method(mlmv)

```
. sem (test1 test2 test3 test4 <- X),  method(mlmv) nolog
Endogenous variables
Measurement:  test1 test2 test3 test4
Exogenous variables
Latent:  X
(output omitted)
Structural equation model  Number of obs  =  500
Estimation method  =  mlmv
Log likelihood  =  -6592.9961
( 1)  [test1]X = 1

<table>
<thead>
<tr>
<th></th>
<th>GIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.         Std. Err.  z  P&gt;</td>
</tr>
<tr>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>test1</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1 (constrained)</td>
</tr>
<tr>
<td>_cons</td>
<td>98.94386</td>
</tr>
<tr>
<td>test2</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1.069952</td>
</tr>
<tr>
<td>_cons</td>
<td>99.84218</td>
</tr>
<tr>
<td>test3</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>.9489025</td>
</tr>
<tr>
<td>_cons</td>
<td>101.0655</td>
</tr>
<tr>
<td>test4</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1.021626</td>
</tr>
<tr>
<td>_cons</td>
<td>99.64509</td>
</tr>
<tr>
<td>var(e.test1)</td>
<td>101.1135</td>
</tr>
<tr>
<td>var(e.test2)</td>
<td>95.45572</td>
</tr>
<tr>
<td>var(e.test3)</td>
<td>95.14847</td>
</tr>
<tr>
<td>var(e.test4)</td>
<td>101.0943</td>
</tr>
<tr>
<td>var(X)</td>
<td>94.04629</td>
</tr>
</tbody>
</table>
```

LR test of model vs. saturated:  $\chi^2(2) = 2.27$, Prob $> \chi^2 = 0.3209$
Notes:

1. The model is now fit using all 500 observations in the dataset.
2. The coefficients for `test3` and `test4`—previously 0.51 and 0.56—are now 0.95 and 1.02.
3. Error variance estimates are now consistent with the true value of 100.
4. Standard errors of path coefficients are mostly smaller than reported in the previous model.
5. `method(mlmv)` requires that the data be MCAR or MAR.
6. `method(mlmv)` requires that the data be multivariate normal.

Fitting the model with the Builder

Use the diagram above for reference.

1. Open the dataset.
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/cfa_missing
   ```

2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Create the measurement component for `X`.
   Select the Add measurement component tool, , and then click in the diagram about one-third of the way down from the top and about halfway in from the left.
   In the resulting dialog box,
   a. change the Latent variable name to X;
   b. select `test1, test2, test3, and test4` by using the Measurement variables control;
   c. select Down in the Measurement direction control;
   d. click on OK.
   If you wish, move the component by clicking on any variable and dragging it.

4. Estimate.
   Click on the Estimate button, , in the Standard Toolbar. In the resulting dialog box,
   a. select the Model tab;
   b. select the Maximum likelihood with missing values radio button;
   c. click on OK.

   You can open a completed diagram in the Builder by typing
   ```
   . webgetsem cfa_missing
   ```

Also see

[SEM] Intro 4 — Substantive concepts
[SEM] `sem option method()` — Specifying method and calculation of VCE
Example 27g — Single-factor measurement model (generalized response)

Description

This is the first example in the g series. The g means that the example focuses exclusively on the \texttt{gsem} command. If you are interested primarily in standard linear SEMs, you may want to skip the remaining examples. If you are especially interested in generalized SEMs, we suggest you read the remaining examples in order.

\texttt{gsem} provides three features not provided by \texttt{sem}: the ability to fit SEMs containing generalized linear response variables, the ability to fit multilevel mixed SEMs, and the ability to fit models with categorical latent variables.

Generalized response variables means that the response variables can be specifications from the generalized linear model (GLM). These include probit, logistic regression, ordered probit and logistic regression, multinomial logistic regression, and more. We use generalized linear response variables in this example.

Multilevel mixed models refer to the simultaneous handling of group-level effects, which can be nested or crossed. Thus you can include unobserved and observed effects for subjects, subjects within group, group within subgroup, . . . , or for subjects, group, subgroup, . . . . See \texttt{SEM} Example 30g for an example of a multilevel model.

Categorical latent variables are latent variables with categories that correspond to groups in the population. The categories are called classes, and we do not observe which individuals belong to which class. Instead, we estimate the probability of being in each class. \texttt{gsem} does not allow categorical latent variables and continuous latent variables (observation level or multilevel) to be present in the same model. See \texttt{SEM} Example 50g for an example with a categorical latent variable.

Below we demonstrate a single-factor measure model with pass/fail (binary outcome) responses rather than continuous responses. We use the following data:

\begin{verbatim}
. use https://www.stata-press.com/data/r16/gsem_1fmm
\end{verbatim}

\begin{verbatim}
(single-factor pass/fail measurement model)
. summarize
\end{verbatim}

\begin{verbatim}
Variable |     Obs  Mean    Std. Dev.    Min   Max
---------+---------------------+---------------------+--------+--------
x1 |     123  .4065041  .4931897        0        1
x2 |     123  .4065041  .4931897        0        1
x3 |     123  .4227642  .4960191        0        1
x4 |     123  .3495935  .4787919        0        1
s4 |     123  690.9837  77.50737        481        885
\end{verbatim}

\begin{verbatim}
. notes
\_dta:
1. Fictional data.
2. The variables x1, x2, x3, and x4 record 1=pass, 0=fail.
3. Pass/fail for x1, x2, x3: score > 100
4. Pass/fail for x4: score > 725
5. Variable s4 contains actual score for test 4.
\end{verbatim}

See \textit{Single-factor measurement models} in \texttt{SEM} \textbf{Intro 5} for background.
Remarks and examples

Remarks are presented under the following headings:

- Single-factor pass/fail measurement model
- Single-factor pass/fail + continuous measurement model
- Fitting the model with the Builder

Single-factor pass/fail measurement model

Below we fit the following model:

\[
\begin{align*}
\text{Pr}(x_1 = 1) &= \Phi(\alpha_1 + X_1 \beta_1) \\
\text{Pr}(x_2 = 1) &= \Phi(\alpha_2 + X_2 \beta_2) \\
\text{Pr}(x_3 = 1) &= \Phi(\alpha_3 + X_3 \beta_3) \\
\text{Pr}(x_4 = 1) &= \Phi(\alpha_4 + X_4 \beta_4)
\end{align*}
\]

where \( \Phi(\cdot) \) is the \( N(0,1) \) cumulative distribution.

One way to think about this is to imagine a test that is scored on a continuous scale. Let’s imagine the scores were \( s_1, s_2, s_3, \) and \( s_4 \) and distributed \( N(\mu_i, \sigma^2_i) \), as test scores often are. Let’s further imagine that for each test, a cutoff \( c_i \) is chosen and the student passes the test if \( s_i \geq c_i \).

If we had the test scores, we would fit this as a linear model. We would posit

\[
s_i = \gamma_i + X_\delta_i + \epsilon_i
\]

where \( \epsilon_i \sim N(0, \sigma_i^2) \). However, we do not have test scores in our data; we have only the pass/fail results \( x_i = 1 \) if \( s_i > c_i \).
So let’s consider the pass/fail problem. The probability that a student passes test \( i \) is determined by the probability that the student scores above the cutoff:

\[
\Pr(s_i > c_i) = \Pr(\gamma_i + X\delta_i + \epsilon_i > c_i)
\]

\[
= \Pr\{\epsilon_i > c_i - (\gamma_i + X\delta_i)\}
\]

\[
= \Pr\{-\epsilon_i \leq -c_i + (\gamma_i + X\delta_i)\}
\]

\[
= \Pr\{-\epsilon_i \leq (\gamma_i - c_i) + X\delta_i\}
\]

\[
= \Pr\{-\epsilon_i/\sigma_i \leq (\gamma_i - c_i)/\sigma_i + X\delta_i/\sigma_i\}
\]

\[
= \Phi\{(\gamma_i - c_i)/\sigma_i + X\delta_i/\sigma_i\}
\]

The last equation is the probit model. In fact, we just derived the probit model, and now we know the relationship between the parameters we will be able to estimate with our pass/fail data: \( \alpha_i \) and \( \beta_i \). We also now know the parameters we could have estimated if we had the continuous test scores: \( \gamma_i \) and \( \delta_i \). The relationship is

\[
\alpha_i = (\gamma_i - c_i)/\sigma_i
\]

\[
\beta_i = \delta_i/\sigma_i
\]

Notice that the right-hand sides of both equations are divided by \( \sigma_i \), the standard deviation of the error term from the linear model for the \( i \)th test score. In pass/fail data, we lose the original scale of the score, and the slope coefficient we will be able to estimate is the slope coefficient from the linear model divided by the standard deviation of the error term. Meanwhile, the intercept we will be able to estimate is just the difference of the continuous model’s intercept and the cutoff for passing the test, sans scale.

The command to fit the model and the results are

\[
gsem (x1 x2 x3 x4 <-X), probit
\]

Fitting fixed-effects model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-329.82091</td>
</tr>
<tr>
<td>1</td>
<td>-329.57665</td>
</tr>
<tr>
<td>2</td>
<td>-329.57664</td>
</tr>
</tbody>
</table>

Refining starting values:

<table>
<thead>
<tr>
<th>Grid node</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-273.75437</td>
</tr>
</tbody>
</table>

Fitting full model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-273.75437</td>
</tr>
<tr>
<td>1</td>
<td>-264.3035</td>
</tr>
<tr>
<td>2</td>
<td>-263.37815</td>
</tr>
<tr>
<td>3</td>
<td>-262.305</td>
</tr>
<tr>
<td>4</td>
<td>-261.69025</td>
</tr>
<tr>
<td>5</td>
<td>-261.42132</td>
</tr>
<tr>
<td>6</td>
<td>-261.35508</td>
</tr>
<tr>
<td>7</td>
<td>-261.3224</td>
</tr>
<tr>
<td>8</td>
<td>-261.3133</td>
</tr>
<tr>
<td>9</td>
<td>-261.30783</td>
</tr>
<tr>
<td>10</td>
<td>-261.30535</td>
</tr>
<tr>
<td>11</td>
<td>-261.30405</td>
</tr>
<tr>
<td>12</td>
<td>-261.30337</td>
</tr>
<tr>
<td>13</td>
<td>-261.30302</td>
</tr>
<tr>
<td>14</td>
<td>-261.30283</td>
</tr>
<tr>
<td>15</td>
<td>-261.30272</td>
</tr>
<tr>
<td>16</td>
<td>-261.30267</td>
</tr>
<tr>
<td>17</td>
<td>-261.30264</td>
</tr>
<tr>
<td>18</td>
<td>-261.30263</td>
</tr>
</tbody>
</table>
Generalized structural equation model Number of obs = 123
Response : x1
Family : Bernoulli
Link : probit
Response : x2
Family : Bernoulli
Link : probit
Response : x3
Family : Bernoulli
Link : probit
Response : x4
Family : Bernoulli
Link : probit
Log likelihood = -261.30263
( 1) [x1]X = 1

|            | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------|--------|-----------|-------|------|----------------------|
| x1         |        |           |       |      |                      |
| X          | 1 (constrained) | -0.3666763 | 0.1896773 | -1.93 | 0.053 | -0.738437 | 0.0050844 |
| _cons      |        |           |       |      |                      |
| x2         |        |           |       |      |                      |
| X          | 1.33293 | 0.4686743  | 2.84  | 0.004 | 0.4143455 | 2.251515 |
| _cons      | -0.4470271 | 0.2372344 | -1.88 | 0.060 | -0.911998 | 0.0179438 |
| x3         |        |           |       |      |                      |
| X          | 0.6040478 | 0.1908343 | 3.17  | 0.002 | 0.2300195 | 0.9780761 |
| _cons      | -0.2276709 | 0.1439342 | -1.58 | 0.114 | -0.5097767 | 0.0544349 |
| x4         |        |           |       |      |                      |
| X          | 9.453342 | 5.151819 | 1.83  | 0.067 | -0.6440375 | 19.55072 |
| _cons      | -4.801027 | 2.518038 | -1.91 | 0.057 | -9.736291 | 1.342372 |
| var(X)     | 2.173451 | 1.044885 | 0.847101 | 5.576536 |

Notes:
1. In the path diagrams, x1, ..., x4 are shown as being family Bernoulli, link probit. On the command line, we just typed probit although we could have typed family(bernoulli) link(probit). In the command language, probit is a synonym for family(bernoulli) link(probit).
2. Variable X is latent exogenous and thus needs a normalizing constraint. The variable is anchored to the first observed variable, x1, and thus the path coefficient is constrained to be 1. See Identification 2: Normalization constraints (anchoring) in SEM Intro 4.
3. The path coefficients for X->x1, X->x2, and X->x3 are 1, 1.33, and 0.60. Meanwhile, the path coefficient for X->x4 is 9.45. This is not unexpected; we at StataCorp generated these fictional data, and we made the x4 effect large and less precisely estimable.

Single-factor pass/fail + continuous measurement model

In the above example, all equations were probit. Different equations within a single SEM can have a different family and link.

Below we refit our model with the continuous measure for test 4 (variable s4) in place of the pass/fail measure (variable x4). We continue to use the pass/fail measures for tests 1, 2, and 3.
Example 27g — Single-factor measurement model (generalized response) 317

. gsem (x1 x2 x3 <-X, probit) (s4<-X)
Fitting fixed-effects model:
Iteration 0: log likelihood = -959.23492
Iteration 1: log likelihood = -959.09499
Iteration 2: log likelihood = -959.09499
Refining starting values:
Grid node 0: log likelihood = -905.14944
Fitting full model:
Iteration 0: log likelihood = -905.14944 (not concave)
Iteration 1: log likelihood = -872.33773
Iteration 2: log likelihood = -869.83144
Iteration 3: log likelihood = -869.69578
Iteration 4: log likelihood = -869.68928
Iteration 5: log likelihood = -869.6892
Generalized structural equation model  Number of obs = 123
Response : x1
Family : Bernoulli
Link : probit
Response : x2
Family : Bernoulli
Link : probit
Response : x3
Family : Bernoulli
Link : probit
Response : s4
Family : Gaussian
Link : identity
Log likelihood = -869.6892
( 1) [x1]X = 1

|        | Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|--------|-------|-----------|-------|-------|---------------------|
| x1     |       |           |       |       |                     |
| _cons  | -.4171085 | .1964736 | -2.12 | 0.034 | -.8021896 -.0320274 |
| x2     |       |           |       |       |                     |
| _cons  | -.4926357 | .2387179 | -2.06 | 0.039 | -.9605142 -.0247573 |
| x3     |       |           |       |       |                     |
| _cons  | -.2942021 | .1575014 | -1.87 | 0.062 | -.6028992 .0144949  |
| s4     |       |           |       |       |                     |
| _cons  | 55.24829 | 12.19904  | 4.53  | 0.000 | 31.3386 79.15798   |
| var(X) | 690.9837 | 6.960106  | 99.28 | 0.000 | 677.3422 704.6253  |
| var(e.s4) | 297.8565 | 408.64    | 0.812856 | 4.230998 | 20.24012 | 4383.299 |

Notes:
1. We obtain similar coefficients for x1, ..., x3.
2. We removed x4 (a pass/fail variable) and substituted s4 (the actual test score). s4 turns out to be more significant than x4. This suggests a poor cutoff was set for “passing” test 4.
3. The log-likelihood values for the two models we have fit are strikingly different: \(-261\) in the previous model and \(-870\) in the current model. The difference has no meaning. Log-likelihood values are dependent on the model specified. We changed the fourth equation from a probit specification to a continuous (linear-regression) specification, and just doing that changes the metric of the log-likelihood function. Comparisons of log-likelihood values are only meaningful when they share the same metric.

**Fitting the model with the Builder**

Use the diagram in *Single-factor pass/fail measurement model* above for reference.

1. Open the dataset.
   
   In the Command window, type
   
   ```plaintext
   . use https://www.stata-press.com/data/r16/gsem_1fmm
   ```

2. Open a new Builder diagram.
   
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation*.

3. Put the Builder in *gsem* mode by clicking on the button.

4. Create the measurement component for \(X\).
   
   Select the Add measurement component tool, \(\mathbb{M}\), and then click in the diagram about one-third of the way down from the top and slightly left of the center.
   
   In the resulting dialog box,
   
   a. change the *Latent variable name* to \(X\);
   b. select \(x1\), \(x2\), \(x3\), and \(x4\) by using the *Measurement variables* control;
   c. check *Make measurements generalized*;
   d. select *Bernoulli, Probit* in the *Family/Link* control;
   e. select *Down* in the *Measurement direction* control;
   f. click on OK.
   
   If you wish, move the component by clicking on any variable and dragging it.

5. Estimate.
   
   Click on the Estimate button, \(\mathbb{E}\), in the Standard Toolbar, and then click on OK in the resulting *GSEM estimation options* dialog box.

6. To fit the model in *Single-factor pass/fail + continuous measurement model*, modify the diagram created in the previous steps.
   
   a. Use the Select tool, \(\mathbb{S}\), to click on the \(x4\) rectangle.
   b. In the Contextual Toolbar, select \(s4\) in the *Variable* control.
   c. In the Contextual Toolbar, select *Gaussian, Identity* in the *Family/Link* control.

7. Estimate.
   
   Click on the Estimate button, \(\mathbb{E}\), in the Standard Toolbar, and then click on OK in the resulting *GSEM estimation options* dialog box.
You can open a completed diagram in the Builder by typing

```
webgetsem gsem_ifmm
```

Also see

- **[SEM] Example 1** — Single-factor measurement model
- **[SEM] Example 28g** — One-parameter logistic IRT (Rasch) model
- **[SEM] Example 29g** — Two-parameter logistic IRT model
- **[SEM] Example 30g** — Two-level measurement model (multilevel, generalized response)
- **[SEM] Example 31g** — Two-factor measurement model (generalized response)
- **[SEM] Example 50g** — Latent class model
- **[SEM] gsem** — Generalized structural equation model estimation command
- **[SEM] Intro 5** — Tour of models
Example 28g — One-parameter logistic IRT (Rasch) model

Description

To demonstrate a one-parameter logistic IRT (Rasch) model, we use the following data:

```
. use https://www.stata-press.com/data/r16/gsem_cfa
(Fictional math abilities data)
. summarize
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>500</td>
<td>10.5</td>
<td>5.772056</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>id</td>
<td>500</td>
<td>50681.71</td>
<td>29081.41</td>
<td>71</td>
<td>100000</td>
</tr>
<tr>
<td>q1</td>
<td>500</td>
<td>.506</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q2</td>
<td>500</td>
<td>.394</td>
<td>.4891242</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q3</td>
<td>500</td>
<td>.534</td>
<td>.4993423</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q4</td>
<td>500</td>
<td>.424</td>
<td>.4946852</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q5</td>
<td>500</td>
<td>.49</td>
<td>.5004006</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q6</td>
<td>500</td>
<td>.434</td>
<td>.4961212</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q7</td>
<td>500</td>
<td>.52</td>
<td>.5001002</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q8</td>
<td>500</td>
<td>.494</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>att1</td>
<td>500</td>
<td>2.946</td>
<td>1.607561</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att2</td>
<td>500</td>
<td>2.948</td>
<td>1.561465</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att3</td>
<td>500</td>
<td>2.84</td>
<td>1.640666</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att4</td>
<td>500</td>
<td>2.91</td>
<td>1.566783</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att5</td>
<td>500</td>
<td>3.086</td>
<td>1.581013</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>test1</td>
<td>500</td>
<td>75.548</td>
<td>5.948653</td>
<td>55</td>
<td>93</td>
</tr>
<tr>
<td>test2</td>
<td>500</td>
<td>80.556</td>
<td>4.976786</td>
<td>65</td>
<td>94</td>
</tr>
<tr>
<td>test3</td>
<td>500</td>
<td>75.572</td>
<td>6.677874</td>
<td>50</td>
<td>94</td>
</tr>
<tr>
<td>test4</td>
<td>500</td>
<td>74.078</td>
<td>8.845587</td>
<td>43</td>
<td>96</td>
</tr>
</tbody>
</table>

. notes

_dta:

1. Fictional data on math ability and attitudes of 500 students from 20 schools.
2. Variables q1–q8 are incorrect/correct (0/1) on individual math questions.
3. Variables att1–att5 are items from a Likert scale measuring each student’s attitude toward math.
4. Variables test1–test4 are test scores from tests of four different aspects of mathematical abilities. Range of scores: 0–100.

These data record results from a fictional instrument measuring mathematical ability. Variables q1 through q8 are the items from the instrument.

For discussions of Rasch models, IRT models, and their extensions, see Embretson and Reise (2000), van der Linden and Hambleton (1997), Skrondal and Rabe-Hesketh (2004), Andrich (1988), Bond and Fox (2015), and Fischer and Molenaar (1995). The standard one-parameter logistic model can be fit using the `irt 1pl` command; see [IRT] `irt 1pl`. This example demonstrates how to fit this model. With _gsem_, we can build on this model to fit many of the extensions to basic IRT models discussed in these books.

See _Item response theory (IRT) models_ in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- 1-PL IRT model with unconstrained variance
- 1-PL IRT model with variance constrained to 1
- Obtaining item characteristic curves
- Fitting the model with the Builder

1-PL IRT model with unconstrained variance

Mechanically speaking, one-parameter logistic (1-PL) IRT models are similar to the probit measurement model we demonstrated in [SEM] Example 27g. The differences are that we will use logit rather than probit and that we will place various constraints on the logit model to obtain results that will allow us to judge the difficulty of the individual questions.

The model we wish to fit is

\[
\text{MathAb} \\
\begin{align*}
q1 & \text{ Bernoulli logit} \\
q2 & \text{ Bernoulli logit} \\
q3 & \text{ Bernoulli logit} \\
q4 & \text{ Bernoulli logit} \\
q5 & \text{ Bernoulli logit} \\
q6 & \text{ Bernoulli logit} \\
q7 & \text{ Bernoulli logit} \\
q8 & \text{ Bernoulli logit}
\end{align*}
\]

In the 1-PL model, we place constraints that all coefficients, the factor loadings, are equal to 1. The negative of the intercept for each question will then represent the difficulty of the question:
Example 28g — One-parameter logistic IRT (Rasch) model

. gsem (MathAb -> (q1-q8)*1), logit
Fitting fixed-effects model:
Iteration 0: log likelihood = -2750.3114
Iteration 1: log likelihood = -2749.3709
Iteration 2: log likelihood = -2749.3708
Refining starting values:
Grid node 0: log likelihood = -2653.2353
Fitting full model:
Iteration 0: log likelihood = -2653.2353
Iteration 1: log likelihood = -2651.2171
Iteration 2: log likelihood = -2650.9117
Iteration 3: log likelihood = -2650.9116
Generalized structural equation model Number of obs = 500
Response : q1
Family : Bernoulli
Link : logit
Response : q2
Family : Bernoulli
Link : logit
Response : q3
Family : Bernoulli
Link : logit
Response : q4
Family : Bernoulli
Link : logit
Response : q5
Family : Bernoulli
Link : logit
Response : q6
Family : Bernoulli
Link : logit
Response : q7
Family : Bernoulli
Link : logit
Response : q8
Family : Bernoulli
Link : logit
Log likelihood = -2650.9116
( 1) [q1]MathAb = 1
( 2) [q2]MathAb = 1
( 3) [q3]MathAb = 1
( 4) [q4]MathAb = 1
( 5) [q5]MathAb = 1
( 6) [q6]MathAb = 1
( 7) [q7]MathAb = 1
( 8) [q8]MathAb = 1
Example 28g — One-parameter logistic IRT (Rasch) model

Coef.

Std. Err.

z

P>|z|

323

[95% Conf. Interval]

q1
MathAb
_cons

1
.0293252

(constrained)
.1047674
0.28

0.780

-.1760152

.2346656

MathAb
_cons

1
-.5025012

(constrained)
.1068768
-4.70

0.000

-.7119759

-.2930264

MathAb
_cons

1
.1607425

(constrained)
.104967
1.53

0.126

-.044989

.3664739

MathAb
_cons

1
-.3574951

(constrained)
.105835
-3.38

0.001

-.564928

-.1500623

MathAb
_cons

1
-.0456599

(constrained)
.1047812
-0.44

0.663

-.2510274

.1597075

MathAb
_cons

1
-.3097521

(constrained)
.1055691
-2.93

0.003

-.5166637

-.1028404

MathAb
_cons

1
.09497

(constrained)
.1048315
0.91

0.365

-.1104959

.300436

MathAb
_cons

1
-.0269104

(constrained)
.1047691
-0.26

0.797

-.232254

.1784332

.6154407

1.02171

q2

q3

q4

q5

q6

q7

q8

var(MathAb)

.7929701

.1025406

Notes:
1. We had to use gsem and not sem to fit this model because the response variables were 0/1 and
not continuous and because we wanted to use logit and not a continuous model.
2. To place the constraints that all coefficients are equal to 1, in the diagram we placed 1s along
the path from the underlying latent factor MathAb to each of the questions. In the command
language, we added @1 to our command:
gsem (MathAb -> (q1-q8)@1), logit

Had we omitted the @1, we would have obtained coefficients about how well each question
measured math ability.
There are several ways we could have asked that the model above be fit. They include the
following:
gsem (MathAb -> q1@1 q2@1 q3@1 q4@1 q5@1 q6@1 q7@1 q8@1), logit
gsem (MathAb -> (q1 q2 q3 q4 q5 q6 q7 q8)@1), logit
gsem (MathAb -> (q1-q8)@1), logit

Similarly, for the shorthand logit, we could have typed family(bernoulli) link(logit).
3. The negative of the reported intercept is proportional to the difficulty of the item. The most
difficult is q2, and the least difficult is q3.


1-PL IRT model with variance constrained to 1

The goal of the 1-PL model is in fact to constrain the loadings to be equal. In the previous model, that was achieved by constraining them to be 1 and letting the variance of the latent variable float. An alternative with perhaps easier-to-interpret results would constrain the variance of the latent variable to be 1—giving it a standard-normal interpretation—and constrain the loadings to be merely equal:

```stata
.gsem (MathAb -> (q1-q8)b), logit var(MathAb@1) nodvheader
Fitting fixed-effects model:
Iteration 0:  log likelihood = -2750.3114
Iteration 1:  log likelihood = -2749.3709
Iteration 2:  log likelihood = -2749.3708
Refining starting values:
Grid node 0:  log likelihood = -2645.8536
Fitting full model:
Iteration 0:  log likelihood = -2656.1973
Iteration 1:  log likelihood = -2650.9139
Iteration 2:  log likelihood = -2650.9116
Iteration 3:  log likelihood = -2650.9116
Generalized structural equation model Number of obs = 500
Log likelihood = -2650.9116
( 1)  [q1]MathAb - [q8]MathAb = 0
( 2)  [q2]MathAb - [q8]MathAb = 0
( 3)  [q3]MathAb - [q8]MathAb = 0
( 4)  [q4]MathAb - [q8]MathAb = 0
( 5)  [q5]MathAb - [q8]MathAb = 0
( 6)  [q6]MathAb - [q8]MathAb = 0
( 7)  [q7]MathAb - [q8]MathAb = 0
( 8)  [/]var(MathAb) = 1

|                | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------|-------|-----------|-------|------|----------------------|
| q1 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | .0293253 | .1047674 | 0.28  | 0.780 | -.1760151 - 0.2346657 |
| q2 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | -.5025011 | .1068768 | -4.70 | 0.000 | -.7119758 - 0.2930264 |
| q3 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | .1607425 | .104967 | 1.53  | 0.126 | -.0449898 - 0.366474 |
| q4 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | -.3574951 | .105835 | -3.38 | 0.001 | -.5649279 - 0.1500622 |
| q5 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | -.0456599 | .1047812 | -0.44 | 0.663 | -.2510273 - 0.1597076 |
| q6 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | -.309752 | .1055691 | -2.93 | 0.003 | -.5166637 - 1.028403 |
| q7 MathAb      | .8904887 | .0575555 | 15.47 | 0.000 | .7776429 - 1.003335 |
| _cons          | .0949701 | .1048315 | 0.91  | 0.365 | -.1104959 - 0.300436 |
```
<table>
<thead>
<tr>
<th>q8</th>
<th>MathAb</th>
<th>.8904887</th>
<th>.0575755</th>
<th>15.47</th>
<th>0.000</th>
<th>.7776429</th>
<th>1.003335</th>
</tr>
</thead>
<tbody>
<tr>
<td>_cons</td>
<td>-0.0269103</td>
<td>.1047691</td>
<td>-0.26</td>
<td>0.797</td>
<td>-.232254</td>
<td>.1784333</td>
<td></td>
</tr>
<tr>
<td>var(MathAb)</td>
<td>1 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:
1. The log-likelihood values of both models is $-2650.9116$. The models are equivalent.
2. Intercepts are unchanged.

**Obtaining item characteristic curves**

Item characteristic curves graph the conditional probability of a particular response given the latent trait. In our case, this simply amounts to graphing the probability of a correct answer against math ability. After estimation, we can obtain the predicted probabilities of a correct answer by typing

```
.predict pr*, pr
(option conditional(ebmeans) assumed)
(using 7 quadrature points)
```

We can obtain the predicted value of the latent variable by typing

```
.predict ability, latent(MathAb)
(option ebmeans assumed)
(using 7 quadrature points)
```

and thus we can obtain the item characteristic curves for all eight questions by typing

```
.twoway line pr1 pr2 pr3 pr4 pr5 pr6 pr7 pr8 ability, sort xlabel(-1.5(.5)1.5)
```

![Item Characteristic Curves](image.png)
A less busy graph might show merely the most difficult and least difficult questions:

```
twoway line pr2 pr3 ability, sort xlabel(-1.5(.5)1.5) legend(col(1))
```

![Graph](image)

The slopes of each curve are identical because we have constrained them to be identical. Thus we just see the shift between difficulties with the lower items having higher levels of difficulty.

### Fitting the model with the Builder

Use the diagram in *1-PL IRT model with unconstrained variance* above for reference.

1. Open the dataset.
   
   In the Command window, type
   
   ```
   . use https://www.stata-press.com/data/r16/gsem_cfa
   ```

2. Open a new Builder diagram.
   
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in `gsem` mode by clicking on the $\mathbb{G}$ button.

4. Create the measurement component for MathAb.
   
   Select the Add measurement component tool, $\mathbb{M}$, and then click in the diagram about one-third of the way down from the top and slightly left of the center.

   In the resulting dialog box,
   
   a. change the **Latent variable name** to MathAb;
   
   b. select q1, q2, q3, q4, q5, q6, q7, and q8 by using the **Measurement variables** control;
   
   c. check **Make measurements generalized**;
   
   d. select Bernoulli, Logit in the **Family/Link** control;
   
   e. select **Down** in the **Measurement direction** control;
   
   f. click on **OK**.

   If you wish, move the component by clicking on any variable and dragging it.
5. Constrain all path coefficients to 1.
   a. Choose the Select tool, 
   b. Click on the path from MathAb to q1. In the Contextual Toolbar, type 1 in the \( \beta \) box and press Enter.
   c. Repeat this process to add the 1 constraint on the paths from MathAb to each of the other measurement variables.

   Click on the Estimate button, \( \mathbf{E} \), in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

7. To fit the model in 1-PL IRT model with variance constrained to 1, change the constraints in the diagram created above.
   a. From the SEM Builder menu, select Estimation > Clear estimates to clear results from the previous model.
   b. Choose the Select tool, \( \mathbf{S} \).
   c. Click on the path from MathAb to q1. In the Contextual Toolbar, type b in the \( \beta \) box and press Enter.
   d. Repeat this process to add the b constraint on the paths from MathAb to each of the other measurement variables.
   e. With \( \mathbf{S} \), click on the oval for MathAb. In the Contextual Toolbar, type 1 in the \( \sigma^2 \) box and press Enter.

8. Estimate again.
   Click on the Estimate button, \( \mathbf{E} \), in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder for the first model by typing
   \[ \text{webgetsem gsem_irt1} \]
You can open a completed diagram in the Builder for the second model by typing
   \[ \text{webgetsem gsem_irt2} \]

References


**Also see**

**[SEM]** Example 27g — Single-factor measurement model (generalized response)

**[SEM]** Example 29g — Two-parameter logistic IRT model

**[SEM]** Intro 5 — Tour of models

**[SEM]** gsem — Generalized structural equation model estimation command

**[SEM]** predict after gsem — Generalized linear predictions, etc.

**[IRT]** irt 1pl — One-parameter logistic model
Example 29g — Two-parameter logistic IRT model

Description

We demonstrate a two-parameter logistic (2-PL) IRT model with the same data used in [SEM] Example 28g:

```
use https://www.stata-press.com/data/r16/gsem_cfa
(Fictional math abilities data)
summarize
```

```
<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>500</td>
<td>10.5</td>
<td>5.772056</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>id</td>
<td>500</td>
<td>50681.71</td>
<td>29081.41</td>
<td>71</td>
<td>10000</td>
</tr>
<tr>
<td>q1</td>
<td>500</td>
<td>.506</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q2</td>
<td>500</td>
<td>.394</td>
<td>.4891242</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q3</td>
<td>500</td>
<td>.534</td>
<td>.4993423</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q4</td>
<td>500</td>
<td>.424</td>
<td>.4946852</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q5</td>
<td>500</td>
<td>.49</td>
<td>.5004006</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q6</td>
<td>500</td>
<td>.434</td>
<td>.4961212</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q7</td>
<td>500</td>
<td>.52</td>
<td>.5001002</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q8</td>
<td>500</td>
<td>.494</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>att1</td>
<td>500</td>
<td>2.946</td>
<td>1.607561</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att2</td>
<td>500</td>
<td>2.948</td>
<td>1.561465</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att3</td>
<td>500</td>
<td>2.84</td>
<td>1.640666</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att4</td>
<td>500</td>
<td>2.91</td>
<td>1.566783</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att5</td>
<td>500</td>
<td>3.086</td>
<td>1.581013</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>test1</td>
<td>500</td>
<td>75.548</td>
<td>5.948653</td>
<td>55</td>
<td>93</td>
</tr>
<tr>
<td>test2</td>
<td>500</td>
<td>80.556</td>
<td>4.976786</td>
<td>65</td>
<td>94</td>
</tr>
<tr>
<td>test3</td>
<td>500</td>
<td>75.572</td>
<td>6.677874</td>
<td>50</td>
<td>94</td>
</tr>
<tr>
<td>test4</td>
<td>500</td>
<td>74.078</td>
<td>8.845587</td>
<td>43</td>
<td>96</td>
</tr>
</tbody>
</table>
```

. notes

_dta:

1. Fictional data on math ability and attitudes of 500 students from 20 schools.
2. Variables q1-q8 are incorrect/correct (0/1) on individual math questions.
3. Variables att1-att5 are items from a Likert scale measuring each student’s attitude toward math.
4. Variables test1-test4 are test scores from tests of four different aspects of mathematical abilities. Range of scores: 0–100.

These data record results from a fictional instrument measuring mathematical ability. Variables q1 through q8 are the items from the instrument.

For discussions of IRT models and their extensions, see Embretson and Reise (2000), van der Linden and Hambleton (1997), Skrondal and Rabe-Hesketh (2004), and Rabe-Hesketh, Skrondal, and Pickles (2004). The two-parameter logistic model can be fit using the `irt 2pl` command; see [IRT] irt 2pl. This example demonstrates how to fit this model. With `gsem`, we can build on this model to fit many of the extensions to basic IRT models discussed in these books.

See Item response theory (IRT) models in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- Fitting the 2-PL IRT model
- Obtaining predicted difficulty and discrimination
- Using coeflegend to obtain the symbolic names of the parameters
- Graphing item characteristic curves
- Fitting the model with the Builder

Fitting the 2-PL IRT model

When we fit the 1-PL model, we commented that it was similar to the probit measure model we demonstrated in [SEM] Example 27g. The 1-PL model differed in that it used logit rather than probit, and it placed constraints on the loadings to judge the difficulty of the individual questions.

The 2-PL model is even more similar to [SEM] Example 27g. We still substitute logit for probit, but we only constrain the variance (the latent variable) to be 1—we leave the loadings unconstrained—and we constrain the variance to be 1 merely to aid interpretation. Compared with the 1-PL example, this time we will measure not just difficulty but discrimination as well.

The model we wish to fit is:

```
MathAb
  q1 Bernoulli logit
  q2 Bernoulli logit
  q3 Bernoulli logit
  q4 Bernoulli logit
  q5 Bernoulli logit
  q6 Bernoulli logit
  q7 Bernoulli logit
  q8 Bernoulli logit
```
The results are

.gsem (MathAb -> q1-q8), logit var(MathAb@1)
Fitting fixed-effects model:
Iteration 0: log likelihood = -2750.3114
Iteration 1: log likelihood = -2749.3709
Iteration 2: log likelihood = -2749.3708
Refining starting values:
Grid node 0: log likelihood = -2645.8536
Fitting full model:
Iteration 0: log likelihood = -2645.8536
Iteration 1: log likelihood = -2637.4315
Iteration 2: log likelihood = -2637.3761
Iteration 3: log likelihood = -2637.3759

Generalized structural equation model
Number of obs = 500
Response  : q1
Family     : Bernoulli
Link       : logit
Response  : q2
Family     : Bernoulli
Link       : logit
Response  : q3
Family     : Bernoulli
Link       : logit
Response  : q4
Family     : Bernoulli
Link       : logit
Response  : q5
Family     : Bernoulli
Link       : logit
Response  : q6
Family     : Bernoulli
Link       : logit
Response  : q7
Family     : Bernoulli
Link       : logit
Response  : q8
Family     : Bernoulli
Link       : logit
Log likelihood = -2637.3759
( 1) ![var(MathAb) = 1](1)

|   | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|---|--------|-----------|-------|------|---------------------|
| q1|        |           |       |      |                     |
| MathAb | 1.466636 | .2488104 | 5.89  | 0.000| .9789765 1.954296  |
| _cons | .0373363 | .1252274 | 0.30  | 0.766| -.208105  .2827776 |
| q2|        |           |       |      |                     |
| MathAb | .5597118 | .1377584 | 4.06  | 0.000| .2897102  .8297134 |
| _cons | -.4613391 | .0989722 | -4.66 | 0.000| -.6553211 -.2673571 |
| q3|        |           |       |      |                     |
| MathAb | .73241  | .1486818 | 4.93  | 0.000| .440999  1.023821 |
| _cons | .1533363 | .1006072 | 1.52  | 0.127| -.0438503 .3505228 |
### Example 29g — Two-parameter logistic IRT model

<table>
<thead>
<tr>
<th>Question</th>
<th>MathAb</th>
<th>Constant</th>
<th>Discrimination Parameter</th>
<th>Difficulty Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>q4</td>
<td>.4839501</td>
<td>.1310028</td>
<td>3.69 0.000</td>
<td>.2271893 .7407109</td>
</tr>
<tr>
<td>_cons</td>
<td>-.3230667</td>
<td>.0957984</td>
<td>-3.37 0.001</td>
<td>-.5108281 -.1353064</td>
</tr>
<tr>
<td>q5</td>
<td>1.232244</td>
<td>.2075044</td>
<td>5.94 0.000</td>
<td>.8255426 1.638945</td>
</tr>
<tr>
<td>_cons</td>
<td>-.0494684</td>
<td>.1163093</td>
<td>-0.43 0.671</td>
<td>-.2774304 .1784937</td>
</tr>
<tr>
<td>q6</td>
<td>.946535</td>
<td>.1707729</td>
<td>5.54 0.000</td>
<td>.6118262 1.281244</td>
</tr>
<tr>
<td>_cons</td>
<td>-.3147231</td>
<td>.1083049</td>
<td>-2.91 0.004</td>
<td>-.5269969 -.1024493</td>
</tr>
<tr>
<td>q7</td>
<td>1.197317</td>
<td>.2029485</td>
<td>5.90 0.000</td>
<td>.7995449 1.595088</td>
</tr>
<tr>
<td>_cons</td>
<td>.1053405</td>
<td>.1152979</td>
<td>0.91 0.361</td>
<td>-.1206393 .3313203</td>
</tr>
<tr>
<td>q8</td>
<td>.8461858</td>
<td>.1588325</td>
<td>5.33 0.000</td>
<td>.5348799 1.157492</td>
</tr>
<tr>
<td>_cons</td>
<td>-.026705</td>
<td>.1034396</td>
<td>-0.26 0.796</td>
<td>-.2294429 .1760329</td>
</tr>
</tbody>
</table>

| var(MathAb) | 1 (constrained) |

### Notes:

1. In the above model, we constrain the variance MathAb to be 1 by typing `var(MathAb@1)`.  
2. Had we not constrained `var(MathAb@1)`, the path coefficient from MathAb to q1 would have automatically constrained to be 1 to set the latent variable's scale. When we applied `var(MathAb@1)`, the automatic constraint was automatically released. Setting the variance of a latent variable is another way of setting its scale.  
3. We set `var(MathAb@1)` to ease interpretation. Our latent variable, MathAb, is now $N(0, 1)$.  
4. Factor loadings, which are the slopes, are estimated above for each question.  
5. The slopes reveal how discriminating each question is in regard to mathematical ability. Question 1 is the most discriminating, and question 4 is the least discriminating.  
6. In the 1-PL model, the negative of the intercept is a measure of difficulty if we constrain the slopes to be equal to each other. To measure difficulty in the 2-PL model, we divide the negative of the intercept by the unconstrained slope. If you do the math, you will discover that question 2 is the most difficult and question 3 is the least difficult. It will be easier, however, merely to continue reading; in the next section, we show an easy way to calculate the discrimination and difficulty for all the questions.

### Obtaining predicted difficulty and discrimination

For each question, discrimination is defined as the question’s slope coefficient.  
For each question, difficulty is defined as the negative of the question’s intercept divided by its slope.
Here is how we quickly obtain all the discrimination and difficulty values in a single, easy-to-read table:

```
. preserve
. drop _all
. set obs 8
   number of observations (_N) was 0, now 8
. generate str question = "q" + strofreal(_n)
. generate diff = .
   (8 missing values generated)
. generate disc = .
   (8 missing values generated)
. forvalues i = 1/8 {
   2.   replace diff = -_b[q'i':_cons] / _b[q'i':MathAb] in 'i'
   3.   replace disc = _b[q'i':MathAb] in 'i'
   4. }
   (1 real change made)
   (1 real change made)
   (1 real change made)
   (1 real change made)
   (1 real change made)
   (1 real change made)
   (1 real change made)
   (1 real change made)
. format diff disc %9.4f
. egen rank_diff = rank(diff)
. egen rank_disc = rank(disc)
. list
```

<table>
<thead>
<tr>
<th>question</th>
<th>diff</th>
<th>disc</th>
<th>rank_d-f</th>
<th>rank_d-c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>q1</td>
<td>-0.0255</td>
<td>1.4666</td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>q2</td>
<td>0.8242</td>
<td>0.5597</td>
<td>8</td>
</tr>
<tr>
<td>3.</td>
<td>q3</td>
<td>-0.2094</td>
<td>0.7324</td>
<td>1</td>
</tr>
<tr>
<td>4.</td>
<td>q4</td>
<td>0.6676</td>
<td>0.4840</td>
<td>7</td>
</tr>
<tr>
<td>5.</td>
<td>q5</td>
<td>0.0401</td>
<td>1.2322</td>
<td>5</td>
</tr>
<tr>
<td>6.</td>
<td>q6</td>
<td>0.3325</td>
<td>0.9465</td>
<td>6</td>
</tr>
<tr>
<td>7.</td>
<td>q7</td>
<td>-0.0880</td>
<td>1.1973</td>
<td>2</td>
</tr>
<tr>
<td>8.</td>
<td>q8</td>
<td>0.0316</td>
<td>0.8462</td>
<td>4</td>
</tr>
</tbody>
</table>

. restore

Notes:

1. Our goal in the Stata code above is to create a dataset containing one observation for each question. The dataset will contain the following variables: question containing q1, q2, ...; diff and disc containing each question’s difficulty and discrimination values; and rank_disc and rank_diff containing the ranks of those discrimination and difficulty values.
2. We first preserved the current data before tossing out the data in memory. Later, after making and displaying our table, we restored the original contents.

3. We then made an 8-observation, 0-variable dataset (set obs 8) and added variables to it. We created string variable question containing q1, q2, ....

4. We were ready to create variables diff and disc. They are defined in terms of estimated coefficients, and we had no idea what the names of those coefficients were. To find out, we typed gsem, coeflegend (output shown below). We quickly learned that the slope coefficients had names like \_b[q1:MathAb], \_b[q2:MathAb], ..., and the intercepts had names like \_b[/q1], \_b[/q2], ....

5. We created new variables diff and disc containing missing values and then created a forvalues loop to fill in the new variables. Notice the odd-looking ‘i’ inside the loop. ‘i’ is the way that you say “substitute the value of (local macro) i here”.

6. We put a display format on new variables diff and disc so that when we listed them, they would be easier to read.

7. We created the rank of each variable by using the egen command.

8. We listed the results. So now you do not have to do the math to see that question 2 is the most difficult (it has rank_diff = 8) and question 3 is the least (it has rank_diff = 1).

9. We typed restore, bringing our original data back into memory and leaving ourselves in a position to continue with this example.

### Using coeflegend to obtain the symbolic names of the parameters

In the section above, we did not retype coefficient values to obtain discrimination and difficulty. After estimation, coefficient values are stored in \_b[name]. To find out what the names are, type gsem, coeflegend. Here are the results:
Graphing item characteristic curves

We showed you the item characteristic curves in [SEM] Example 28g, so we will show them to you again. Graphs of item characteristic curves plot the probability of a correct answer against the latent trait, which in this case is math ability.

We obtain the probabilities of a correct answer (the values of the latent variable) just as we did previously,

```stata
. predict pr2pl*, pr (option conditional(ebmeans) assumed) (using 7 quadrature points)
. predict ability2pl, latent(MathAb) (option ebmeans assumed) (using 7 quadrature points)
```

<table>
<thead>
<tr>
<th>Coef. Legend</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
<th>q4</th>
<th>q5</th>
<th>q6</th>
<th>q7</th>
<th>q8</th>
</tr>
</thead>
<tbody>
<tr>
<td>_cons</td>
<td>.0373363 _b[q1:_cons]</td>
<td>-.4613391 _b[q2:_cons]</td>
<td>.1533363 _b[q3:_cons]</td>
<td>-.3230667 _b[q4:_cons]</td>
<td>-.0494684 _b[q5:_cons]</td>
<td>-.3147231 _b[q6:_cons]</td>
<td>.1053405 _b[q7:_cons]</td>
<td>-.026705 _b[q8:_cons]</td>
</tr>
<tr>
<td>var(MathAb)</td>
<td>1 _b[/var(MathAb)]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
and we graph the curves just as we did previously, too. Here are all eight curves on one graph:

```
    . twoway line pr2pl* ability2pl, sort xlabel(-1.5(.5)1.5) legend(col(1))
```

In [SEM] Example 28g, we showed a graph for the most and least difficult questions. This time we show a graph for the most and least discriminating questions:

```
    . twoway line pr2pl1 pr2pl4 ability2pl, sort xlabel(-1.5(.5)1.5) legend(col(1))
```

Here the curves are not parallel because the discrimination has not been constrained to be equal across the questions. Question 1 has a steeper slope, so it is more discriminating.

**Fitting the model with the Builder**

Use the diagram in Fitting the 2-PL IRT model above for reference.

1. Open the dataset.
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/gsem_cfa
   ```
2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in **gsem** mode by clicking on the $g$ button.

4. Create the measurement component for **MathAb**.
   Select the Add measurement component tool, $\mathbb{W}$, and then click in the diagram about one-third of the way down from the top and slightly left of the center.
   In the resulting dialog box,
   a. change the **Latent variable name** to **MathAb**;
   b. select $q_1$, $q_2$, $q_3$, $q_4$, $q_5$, $q_6$, $q_7$, and $q_8$ by using the **Measurement variables** control;
   c. check **Make measurements generalized**;
   d. select **Bernoulli, Logit** in the **Family/Link** control;
   e. select **Down** in the **Measurement direction** control;
   f. click on **OK**.
   If you wish, move the component by clicking on any variable and dragging it.

5. Constrain the variance of **MathAb** to 1.
   a. Choose the Select tool, $\mathbb{I}$.
   b. Click on the oval for **MathAb**. In the Contextual Toolbar, type 1 in the $\mathbb{I}$ box and press Enter.

   Click on the **Estimate** button, $\mathbb{E}$, in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

   You can open a completed diagram in the Builder by typing
   ```
   .webgetsem gsem_irt3
   ```

**References**


Also see

[SEM] Example 27g — Single-factor measurement model (generalized response)

[SEM] Example 28g — One-parameter logistic IRT (Rasch) model

[SEM] Intro 5 — Tour of models

[SEM] gsem — Generalized structural equation model estimation command

[SEM] predict after gsem — Generalized linear predictions, etc.

[IRT] irt 2pl — Two-parameter logistic model
Example 30g — Two-level measurement model (multilevel, generalized response)

Description

We demonstrate a multilevel measurement model with the same data used in [SEM] Example 29g:

```
. use https://www.stata-press.com/data/r16/gsem_cfa
    (Fictional math abilities data)
. summarize
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>500</td>
<td>10.5</td>
<td>5.772056</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>id</td>
<td>500</td>
<td>50681.71</td>
<td>29081.41</td>
<td>71</td>
<td>10000</td>
</tr>
<tr>
<td>q1</td>
<td>500</td>
<td>.506</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q2</td>
<td>500</td>
<td>.394</td>
<td>.4891242</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q3</td>
<td>500</td>
<td>.534</td>
<td>.4993423</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q4</td>
<td>500</td>
<td>.424</td>
<td>.4946852</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q5</td>
<td>500</td>
<td>.49</td>
<td>.5004006</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q6</td>
<td>500</td>
<td>.434</td>
<td>.4961212</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q7</td>
<td>500</td>
<td>.52</td>
<td>.5001002</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q8</td>
<td>500</td>
<td>.494</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>att1</td>
<td>500</td>
<td>2.946</td>
<td>1.607561</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att2</td>
<td>500</td>
<td>2.948</td>
<td>1.561465</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att3</td>
<td>500</td>
<td>2.84</td>
<td>1.640666</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att4</td>
<td>500</td>
<td>2.91</td>
<td>1.566783</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att5</td>
<td>500</td>
<td>3.086</td>
<td>1.581013</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>test1</td>
<td>500</td>
<td>75.548</td>
<td>5.948653</td>
<td>55</td>
<td>93</td>
</tr>
<tr>
<td>test2</td>
<td>500</td>
<td>80.556</td>
<td>4.976786</td>
<td>65</td>
<td>94</td>
</tr>
<tr>
<td>test3</td>
<td>500</td>
<td>75.572</td>
<td>6.677874</td>
<td>50</td>
<td>94</td>
</tr>
<tr>
<td>test4</td>
<td>500</td>
<td>74.078</td>
<td>8.845587</td>
<td>43</td>
<td>96</td>
</tr>
</tbody>
</table>

. notes

_dta:
1. Fictional data on math ability and attitudes of 500 students from 20 schools.
2. Variables q1–q8 are incorrect/correct (0/1) on individual math questions.
3. Variables att1–att5 are items from a Likert scale measuring each student’s attitude toward math.
4. Variables test1–test4 are test scores from tests of four different aspects of mathematical abilities. Range of scores: 0–100.

These data record results from a fictional instrument measuring mathematical ability. Variables q1 through q8 are the items from the instrument.

For discussions of multilevel measurement models, including extensions beyond the example we present here, see Mehta and Neale (2005) and Skrondal and Rabe-Hesketh (2004).

See Single-factor measurement models and Multilevel mixed-effects models in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- Fitting the two-level model
- Fitting the variance-components model
- Fitting the model with the Builder

Fitting the two-level model

We extend the measurement model fit in [SEM] Example 29g to better account for our (fictional) data. In the data, students are nested within school, but we have ignored that so far. In this example, we include a latent variable at the school level to account for possible school-by-school effects.

The model we wish to fit is

\[
\text{MathAb} \\
\text{q1 Bernoulli logit q2 Bernoulli logit q3 Bernoulli logit q4 Bernoulli logit q5 Bernoulli logit q6 Bernoulli logit q7 Bernoulli logit q8 Bernoulli logit}
\]

\text{school}_1

The double-ringed \text{school}_1 is new. That new component of the path diagram is saying, “I am a latent variable at the school level—meaning I am constant within school and vary across schools—and I correspond to a latent variable named M1”; see Specifying generalized SEMs: Multilevel mixed effects (2 levels) in [SEM] Intro 2. This new variable will account for the effect, if any, of the identity of the school.

To fit this model without this new, school-level component in it, we would type

\[
\text{. gsem (MathAb -> q1-q8), logit}
\]
To include the new school-level component, we add $M1\text{[school]}$ to the exogenous variables:

```
. gsem (MathAb $M1\text{[school]}$ -> q1-q8), logit
```

Fitting fixed-effects model:

```
Iteration 0:  log likelihood = -2750.3114
Iteration 1:  log likelihood = -2749.3709
Iteration 2:  log likelihood = -2749.3708
```

Refining starting values:

```
Grid node 0:  log likelihood = -2649.0033
```

Fitting full model:

```
Iteration 0:  log likelihood = -2649.0033  (not concave)
Iteration 1:  log likelihood = -2645.0613  (not concave)
Iteration 2:  log likelihood = -2641.9755  (not concave)
Iteration 3:  log likelihood = -2634.3857
Iteration 4:  log likelihood = -2631.1111
Iteration 5:  log likelihood = -2630.7898
Iteration 6:  log likelihood = -2630.2477
Iteration 7:  log likelihood = -2630.2402
Iteration 8:  log likelihood = -2630.2074
Iteration 9:  log likelihood = -2630.2063
Iteration 10: log likelihood = -2630.2063
```

Generalized structural equation model

```
Response  : q1
Family     : Bernoulli
Link       : logit
Response  : q2
Family     : Bernoulli
Link       : logit
Response  : q3
Family     : Bernoulli
Link       : logit
Response  : q4
Family     : Bernoulli
Link       : logit
Response  : q5
Family     : Bernoulli
Link       : logit
Response  : q6
Family     : Bernoulli
Link       : logit
Response  : q7
Family     : Bernoulli
Link       : logit
Response  : q8
Family     : Bernoulli
Link       : logit
```

Log likelihood = -2630.2063

( 1)  [q1]$M1\text{[school]} = 1$
( 2)  [q2]MathAb = 1
### Example 30g — Two-level measurement model (multilevel, generalized response)

|    | Coef.   | Std. Err. | z      | P>|z| | [95% Conf. Interval] |
|----|---------|-----------|--------|-----|---------------------|
|q1  | M1[school] | 1 (constrained) |       |     |                     |
|    | MathAb  | 2.807515  | .9468682 | 2.97| 0.003               |
|    | _cons   | .0388021  | .1608489 | 0.24| 0.809               |
|    |         | .9516878  | 4.663343 |
|q2  | M1[school] | .6673925  | .3058328 | 2.18| 0.029               |
|    | MathAb  | 1 (constrained) |       |     |                     |
|    | _cons   | -.4631159 | .1201227 | -3.86| 0.000               |
|    |         | -.698552  | -.227679 |
|q3  | M1[school] | .3555867  | .3043548 | 1.17| 0.243               |
|    | MathAb  | 1.455529  | .5187786 | 2.81| 0.005               |
|    | _cons   | .1537831  | .1070288 | 1.44| 0.151               |
|    |         | .4387416  | 2.472316 |
|q4  | M1[school] | .7073241  | .3419273 | 2.07| 0.039               |
|    | MathAb  | .8420897  | .3528195 | 2.39| 0.017               |
|    | _cons   | -.3282735 | .1202088 | -2.71| 0.007               |
|    |         | -.560874  | -.089668 |
|q5  | M1[school] | .7295553  | .3330652 | 2.19| 0.028               |
|    | MathAb  | 2.399529  | .8110973 | 2.96| 0.003               |
|    | _cons   | -.0488674 | .1378015 | -0.35| 0.723               |
|    |         | -.3189533 | .2212185 |
|q6  | M1[school] | .484903   | .2844447 | 1.70| 0.088               |
|    | MathAb  | 1.840627  | .5934017 | 3.10| 0.002               |
|    | _cons   | -.3139302 | .1186624 | -2.65| 0.008               |
|    |         | -.5465042 | -.081356 |
|q7  | M1[school] | .3677241  | .2735779 | 1.34| 0.179               |
|    | MathAb  | 2.444023  | .8016872 | 3.05| 0.002               |
|    | _cons   | .1062164  | .1220796 | 0.87| 0.384               |
|    |         | .8727449  | 4.015301 |
|q8  | M1[school] | .5851299  | .3449508 | 1.70| 0.090               |
|    | MathAb  | 1.606287  | .5367614 | 2.99| 0.003               |
|    | _cons   | -.0261962 | .1189835 | -0.22| 0.826               |
|    |         | -.2593995 | .2070071 |
|    |         | .052558   | .8561121 |
|    |         | .0825055  | .7342217 |

| var(M1[school]) | .2121216  | .1510032 | .052558 | .8561121 |
| var(MathAb)     | .2461246  | .1372513 | .0825055 | .7342217 |
Notes:

1. The variance of $M_{1[school]}$ is estimated to be 0.21.

2. So how important is $M_{1[school]}$? The variance of MathAb is estimated to be 0.25, so math ability and school have roughly the same variance, and both of course have mean 0. The math ability coefficients, meanwhile, are larger—often much larger—than the school coefficients in every case, so math ability is certainly more important than school in explaining whether questions were answered correctly. At this point, we are merely exploring the magnitude of effect.

3. You could also include a school-level latent variable for each question. For instance, you could type

   `. gsem (MathAb M1[school] N1[school] -> q1) ///
   (MathAb M1[school] N2[school] -> q2) ///
   (MathAb M1[school] N3[school] -> q3) ///
   (MathAb M1[school] N4[school] -> q4) ///
   (MathAb M1[school] N5[school] -> q5) ///
   (MathAb M1[school] N6[school] -> q6) ///
   (MathAb M1[school] N7[school] -> q7) ///
   (MathAb M1[school] N8[school] -> q8), logit`

   You will sometimes see such effects included in multilevel measurement models in theoretical discussions of models. Be aware that estimation of models with many latent variables is problematic, requiring both time and luck.

Fitting the variance-components model

In a variance-components model, school would affect math ability which would affect correctness of answers to questions. The model might be drawn like this:
The above is a great way to draw the model. Sadly, `gsem` cannot understand it. The problem from `gsem`’s perspective is that one latent variable is affecting another and the two latent variables are at different levels.

So we have to draw the model differently:

The models may look different, but constraining the coefficients along the paths from math ability and from school to each question is identical in effect to the model above.
The result of fitting the model is

```
.gsem (MathAb M1[school] ->
    q101 q20c2 q30c3 q40c4 q50c5 q60c6 q70c7 q80c8), logit
Fitting fixed-effects model:
Iteration 0:  log likelihood = -2750.3114
Iteration 1:  log likelihood = -2749.3709
Iteration 2:  log likelihood = -2749.3708
Refining starting values:
Grid node 0:  log likelihood = -2642.8248
Fitting full model:
Iteration 0:  log likelihood = -2651.7239  (not concave)
Iteration 1:  log likelihood = -2644.4937
Iteration 2:  log likelihood = -2634.92
Iteration 3:  log likelihood = -2633.9336
Iteration 4:  log likelihood = -2633.5924
Iteration 5:  log likelihood = -2633.5922
Generalized structural equation model  Number of obs =  500
(output omitted)
Log likelihood = -2633.5922
( 1)  [q1]M1[school] = 1
( 2)  [q1]MathAb = 1
( 3)  [q2]M1[school] - [q2]MathAb = 0
( 4)  [q3]M1[school] - [q3]MathAb = 0
( 9)  [q8]M1[school] - [q8]MathAb = 0

|                | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|----------------|-------|-----------|-------|------|---------------------|
| q1              |       |           |       |      |                     |
| M1[school]      | 1     | (constrained) |
| MathAb          | 1     | (constrained) |
| _cons           | 0.0385522 | 0.1556214 | 0.25  | 0.804 | -0.2664601 - 0.3435646 |

| q2              |       |           |       |      |                     |
| M1[school]      | 0.3876281 | 0.1156823 | 3.35  | 0.001 | 0.1608951 - 0.6143612 |
| MathAb          | 0.3876281 | 0.1156823 | 3.35  | 0.001 | 0.1608951 - 0.6143612 |
| _cons           | -0.4633143 | 0.1055062 | -4.39 | 0.000 | -0.6701028 - -0.2565259 |

| q3              |       |           |       |      |                     |
| M1[school]      | 0.4871164 | 0.1295515 | 3.76  | 0.000 | 0.2332001 - 0.7410328 |
| MathAb          | 0.4871164 | 0.1295515 | 3.76  | 0.000 | 0.2332001 - 0.7410328 |
| _cons           | 0.1533212 | 0.1098068 | 1.40  | 0.163 | -0.0618962 - 0.3685386 |

| q4              |       |           |       |      |                     |
| M1[school]      | 0.3407151 | 0.1058542 | 3.22  | 0.001 | 0.1332446 - 0.5481856 |
| MathAb          | 0.3407151 | 0.1058542 | 3.22  | 0.001 | 0.1332446 - 0.5481856 |
| _cons           | -0.3246936 | 0.1011841 | -3.21 | 0.001 | -0.5230108 - -0.1263763 |
### Example 30g — Two-level measurement model (multilevel, generalized response)

<table>
<thead>
<tr>
<th>Question</th>
<th>Parameter</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>z</th>
<th>Prob &gt;</th>
<th>Lower 95%</th>
<th>Upper 95%</th>
</tr>
</thead>
<tbody>
<tr>
<td>q5</td>
<td>M1[school]</td>
<td>.8327426</td>
<td>.1950955</td>
<td>4.27</td>
<td>0.000</td>
<td>.4503624</td>
<td>1.215123</td>
</tr>
<tr>
<td></td>
<td>MathAb</td>
<td>.8327426</td>
<td>.1950955</td>
<td>4.27</td>
<td>0.000</td>
<td>.4503624</td>
<td>1.215123</td>
</tr>
<tr>
<td></td>
<td>_cons</td>
<td>-.0490579</td>
<td>.1391324</td>
<td>-0.35</td>
<td>0.724</td>
<td>-.3217524</td>
<td>.2236365</td>
</tr>
<tr>
<td>q6</td>
<td>M1[school]</td>
<td>.6267415</td>
<td>.1572247</td>
<td>3.99</td>
<td>0.000</td>
<td>.3185868</td>
<td>.9348962</td>
</tr>
<tr>
<td></td>
<td>MathAb</td>
<td>.6267415</td>
<td>.1572247</td>
<td>3.99</td>
<td>0.000</td>
<td>.3185868</td>
<td>.9348962</td>
</tr>
<tr>
<td></td>
<td>_cons</td>
<td>-.3135398</td>
<td>.1220389</td>
<td>-2.57</td>
<td>0.010</td>
<td>-.5527317</td>
<td>-.074348</td>
</tr>
<tr>
<td>q7</td>
<td>M1[school]</td>
<td>.7660343</td>
<td>.187918</td>
<td>4.08</td>
<td>0.000</td>
<td>.3977219</td>
<td>1.134347</td>
</tr>
<tr>
<td></td>
<td>MathAb</td>
<td>.7660343</td>
<td>.187918</td>
<td>4.08</td>
<td>0.000</td>
<td>.3977219</td>
<td>1.134347</td>
</tr>
<tr>
<td></td>
<td>_cons</td>
<td>.1039102</td>
<td>.1330652</td>
<td>0.78</td>
<td>0.435</td>
<td>-.1568927</td>
<td>.3647131</td>
</tr>
<tr>
<td>q8</td>
<td>M1[school]</td>
<td>.5600833</td>
<td>.1416542</td>
<td>3.95</td>
<td>0.000</td>
<td>.2824462</td>
<td>.8377203</td>
</tr>
<tr>
<td></td>
<td>MathAb</td>
<td>.5600833</td>
<td>.1416542</td>
<td>3.95</td>
<td>0.000</td>
<td>.2824462</td>
<td>.8377203</td>
</tr>
<tr>
<td></td>
<td>_cons</td>
<td>-.0264193</td>
<td>.1150408</td>
<td>-0.23</td>
<td>0.818</td>
<td>-.2518951</td>
<td>.1990565</td>
</tr>
</tbody>
</table>

|  | var(M1[school]) | .1719347 | .1150138  | .0463406 | .6379187 |
|  | var(MathAb)     | 2.062489 | .6900045  | 1.070589 | 3.973385 |

1. Note that for each question, the coefficient on MathAb is identical to the coefficient on M1[school].
2. We estimate separate variances for M1[school] and MathAb. They are 0.17 and 2.06. Now that the coefficients are the same on school and ability, we can directly compare these variances. We see that math ability has a much larger affect than does school.

### Fitting the model with the Builder

Use the diagram in *Fitting the two-level model* above for reference.

1. Open the dataset.
   In the Command window, type
   ```stata
   . use https://www.stata-press.com/data/r16/gsem_cfa
   ```

2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in `gsem` mode by clicking on the $\mathcal{G}$ button.

4. Create the measurement component for MathAb.
   Select the Add measurement component tool, $\mathcal{V}$, and then click in the diagram about one-fourth of the way down from the top and slightly left of the center.
   In the resulting dialog box,
   a. change the *Latent variable name* to MathAb;
   b. select q1, q2, q3, q4, q5, q6, q7, and q8 by using the *Measurement variables* control;
c. check Make measurements generalized;
d. select Bernoulli, Logit in the Family/Link control;
e. select Down in the Measurement direction control;
f. click on OK.

If you wish, move the component by clicking on any variable and dragging it.

5. Create the school-level latent variable.
   a. Select the Add multilevel latent variable tool, ⬧, and click about one-fourth of the way up from the bottom and slightly left of the center.
   b. In the Contextual Toolbar, click on the button.
   c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting school > Observations in the next control.
   d. Specify M1 as the Base name.
   e. Click on OK.

6. Create the factor-loading paths for the multilevel latent variable.
   a. Select the Add path tool, ←.
   b. Click in the top-left quadrant of the double oval for school1 (it will highlight when you hover over it), and drag a path to the bottom of the q1 rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the ← tool, draw paths from school1 to each of the remaining rectangles.

7. Clean up paths.

   If you do not like where a path has been connected to its variables, use the Select tool, ⬹, to click on the path, and then simply click on where it connects to a rectangle or oval and drag the endpoint.

8. Estimate.

   Click on the Estimate button, ➡, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

9. To fit the model in Fitting the variance-components model, add constraints to the diagram created above.
   a. From the SEM Builder menu, select Estimation > Clear estimates to clear results from the previous model.
   b. Choose the Select tool, ⬹.
   c. Click on the path from MathAb to q1. In the Contextual Toolbar, type 1 in the $\beta$ box and press Enter.
   d. Click on the path from school1 to q1. In the Contextual Toolbar, type 1 in the $\beta$ box and press Enter.
   c. Click on the path from MathAb to q2. In the Contextual Toolbar, type c2 in the $\beta$ box and press Enter.
   d. Click on the path from school1 to q2. In the Contextual Toolbar, type c2 in the $\beta$ box and press Enter.
e. Repeat this process to add the c3 constraint on both paths to q3, the c4 constraint on both paths to q4, \ldots, and the c8 constraint on both paths to q8.

10. Estimate again.

Click on the **Estimate** button, in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

You can open a completed diagram in the Builder for the first model by typing

```
.webgetsem gsem_mlcfa1
```

You can open a completed diagram in the Builder for the second model by typing

```
.webgetsem gsem_mlcfa2
```

References


Also see

- [SEM] **Example 27g** — Single-factor measurement model (generalized response)
- [SEM] **Example 29g** — Two-parameter logistic IRT model
- [SEM] **Intro 5** — Tour of models
- [SEM] **gsem** — Generalized structural equation model estimation command
We demonstrate a two-factor generalized linear measurement model with the same data used in [SEM] Example 29g:

```
. use https://www.stata-press.com/data/r16/gsem_cfa
(Fictional math abilities data)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_cfa.dta
obs: 500 Fictional math abilities data
vars: 19 21 Mar 2018 10:38
(_dta has notes)

| variable name | storage | display | value
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>long</td>
<td>%9.0g</td>
<td></td>
</tr>
<tr>
<td>q1</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q1 correct</td>
</tr>
<tr>
<td>q2</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q2 correct</td>
</tr>
<tr>
<td>q3</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q3 correct</td>
</tr>
<tr>
<td>q4</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q4 correct</td>
</tr>
<tr>
<td>q5</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q5 correct</td>
</tr>
<tr>
<td>q6</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q6 correct</td>
</tr>
<tr>
<td>q7</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q7 correct</td>
</tr>
<tr>
<td>q8</td>
<td>byte</td>
<td>%9.0g</td>
<td>result q8 correct</td>
</tr>
<tr>
<td>att1</td>
<td>float</td>
<td>%26.0g</td>
<td>agree Skills taught in math class will help me get a better job.</td>
</tr>
<tr>
<td>att2</td>
<td>float</td>
<td>%26.0g</td>
<td>agree Math is important in everyday life</td>
</tr>
<tr>
<td>att3</td>
<td>float</td>
<td>%26.0g</td>
<td>agree Working math problems makes me anxious.</td>
</tr>
<tr>
<td>att4</td>
<td>float</td>
<td>%26.0g</td>
<td>agree Math has always been my worst subject.</td>
</tr>
<tr>
<td>att5</td>
<td>float</td>
<td>%26.0g</td>
<td>agree I am able to learn new math concepts easily.</td>
</tr>
<tr>
<td>test1</td>
<td>byte</td>
<td>%9.0g</td>
<td>Score, math test 1</td>
</tr>
<tr>
<td>test2</td>
<td>byte</td>
<td>%9.0g</td>
<td>Score, math test 2</td>
</tr>
<tr>
<td>test3</td>
<td>byte</td>
<td>%9.0g</td>
<td>Score, math test 3</td>
</tr>
<tr>
<td>test4</td>
<td>byte</td>
<td>%9.0g</td>
<td>Score, math test 4</td>
</tr>
</tbody>
</table>
```

Sorted by:

. notes

_dta:
1. Fictional data on math ability and attitudes of 500 students from 20 schools.
2. Variables q1-q8 are incorrect/correct (0/1) on individual math questions.
3. Variables att1-att5 are items from a Likert scale measuring each student’s attitude toward math.
4. Variables test1-test4 are test scores from tests of four different aspects of mathematical abilities. Range of scores: 0-100.
These data record results from a fictional instrument measuring mathematical ability. Variables q1 through q8 are the items from the instrument.

In this example, we will also be using variables att1 through att5. These are five Likert-scale questions measuring each student’s attitude toward math.

See *Multiple-factor measurement models* in [SEM] Intro 5 for background.

**Remarks and examples**

Remarks are presented under the following headings:

* Fitting the two-factor model
* Fitting the model with the Builder

**Fitting the two-factor model**

We extend the measurement model fit in [SEM] Example 29g from one factor, math ability, to two factors, math ability and attitude. The model we wish to fit is

In this model, mathematical ability affects the correctness of the answers to the items just as previously. The new component, attitude toward mathematics, is correlated with math ability. We expect this correlation to be positive, but that is yet to be determined.

What is important about the attitudinal questions is that the responses are ordinal, that is, the ordering of the possible answers is significant. In other cases, we might have a categorical variable taking on, say, five values; even if the values are 1, 2, 3, 4, and 5, there is no case in which answer 5 is greater than answer 4, answer 4 is greater than answer 3, and so on.
For our attitude measures, however, response 5 signifies strong agreement with a statement and 1 signifies strong disagreement. We handle the ordinal property by specifying that the attitudinal responses are family ordinal, link logit, also known as ordered logit or ordinal logistic regression, and also known in Stata circles as ologit.

In the command language, to fit a one-factor measurement model with math ability, we would type

```
gsem (MathAb -> q1-q8), logit
```

To include the second factor, attitude correlated with math ability, we would type

```
gsem (MathAb -> q1-q8, logit) ///
     (MathAtt -> att1-att5, ologit)
```

The covariance between MathAtt and MathAb does not even appear in the command! That is because latent exogenous variables are assumed to be correlated in the command language unless you specify otherwise; in path diagrams, such variables are correlated only if a curved path is drawn between them.

There is another, minor difference in syntax between the one-factor and two-factor models that is worth your attention. Notice that the `logit` was outside the parentheses in the command to fit the one-factor model, but it is inside the parentheses in the command to fit the two-factor model. Actually, `logit` could have appeared inside the parentheses to fit the one-factor model. When options appear inside parentheses, they affect only what is specified inside the parentheses. When they appear outside parentheses, they affect all parenthetical specifications.

To obtain the estimates of the two-factor model, we type

```
  . gsem (MathAb -> q1-q8, logit)
     > (MathAtt -> att1-att5, ologit)

Fitting fixed-effects model:
  Iteration 0:  log likelihood = -6629.7253
  Iteration 1:  log likelihood = -6628.7848
  Iteration 2:  log likelihood = -6628.7848
Refining starting values:
  Grid node 0:  log likelihood = -6457.4584
Fitting full model:
  Iteration 0:  log likelihood = -6457.4584
  Iteration 1:  log likelihood = -6437.9594
  Iteration 2:  log likelihood = -6400.2731
  Iteration 3:  log likelihood = -6396.3795
  Iteration 4:  log likelihood = -6394.5787
  Iteration 5:  log likelihood = -6394.4019
  Iteration 6:  log likelihood = -6394.3923
  Iteration 7:  log likelihood = -6394.3923

Generalized structural equation model  Number of obs = 500
Response :  q1
  Family   :  Bernoulli
     Link   :  logit
Response :  q2
  Family   :  Bernoulli
     Link   :  logit
Response :  q3
  Family   :  Bernoulli
     Link   :  logit
Response :  q4
  Family   :  Bernoulli
     Link   :  logit
Example 31g — Two-factor measurement model (generalized response)

<table>
<thead>
<tr>
<th>Response</th>
<th>Family</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>q5</td>
<td>Bernoulli</td>
<td>logit</td>
</tr>
<tr>
<td>q6</td>
<td>Bernoulli</td>
<td>logit</td>
</tr>
<tr>
<td>q7</td>
<td>Bernoulli</td>
<td>logit</td>
</tr>
<tr>
<td>q8</td>
<td>Bernoulli</td>
<td>logit</td>
</tr>
<tr>
<td>att1</td>
<td>ordinal</td>
<td>logit</td>
</tr>
<tr>
<td>att2</td>
<td>ordinal</td>
<td>logit</td>
</tr>
<tr>
<td>att3</td>
<td>ordinal</td>
<td>logit</td>
</tr>
<tr>
<td>att4</td>
<td>ordinal</td>
<td>logit</td>
</tr>
<tr>
<td>att5</td>
<td>ordinal</td>
<td>logit</td>
</tr>
</tbody>
</table>

Log likelihood = -6394.3923

(1) [q1] MathAb = 1
(2) [att1] MathAtt = 1

|     | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-----|-------|-----------|-------|------|----------------------|
| q1  | MathAb | 1 (constrained) | .0446118 | .1272964 | 0.35 | 0.726 | -.2048845 .2941082 |
|     | _cons  |           |       |       |         |       |                     |
| q2  | MathAb  | .3446081 | .1050264 | 3.28 | 0.001 | .1387601 .5504562 |
|     | _cons  | -.4572215 | .0979965 | -4.67 | 0.000 | -.6492911 -.265152 |
| q3  | MathAb  | .5445245 | .1386993 | 3.93 | 0.000 | .272679 .8163701 |
|     | _cons  | .1591406 | .1033116 | 1.54 | 0.123 | -.0433464 .3616276 |
| q4  | MathAb  | .2858874 | .0948553 | 3.01 | 0.003 | .0999743 .4718004 |
|     | _cons  | -.3196648 | .0947684 | -3.37 | 0.001 | -.5054075 -.139222 |
| q5  | MathAb  | .8174803 | .1867024 | 4.38 | 0.000 | .4515504 1.18341 |
|     | _cons  | -.04543 | .116575 | -0.39 | 0.697 | -.2739127 .1830527 |
| q6  | MathAb  | .6030448 | .1471951 | 4.10 | 0.000 | .3145478 .8915419 |
|     | _cons  | -.309992 | .1070853 | -2.89 | 0.004 | -.5198753 -.1001086 |
### Example 31g — Two-factor measurement model (generalized response)

<table>
<thead>
<tr>
<th>q7</th>
<th>MathAb</th>
<th>_cons</th>
<th>0.72084</th>
<th>0.1713095</th>
<th>4.21</th>
<th>0.000</th>
<th>0.3850796</th>
<th>1.056601</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.1047265</td>
<td>0.116494</td>
<td>0.94</td>
<td>0.348</td>
<td>-0.1141023</td>
<td>0.3235552</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>q8</th>
<th>MathAb</th>
<th>_cons</th>
<th>0.5814761</th>
<th>0.1426727</th>
<th>4.08</th>
<th>0.000</th>
<th>0.3018428</th>
<th>0.8611094</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.0250442</td>
<td>0.1045134</td>
<td>-0.24</td>
<td>0.811</td>
<td>-0.2298868</td>
<td>0.1797983</td>
</tr>
</tbody>
</table>

| att1 | MathAtt | 1 (constrained) |
|      |        |               |

| att2 | MathAtt |        | 0.3788714 | 0.0971223 | 3.90 | 0.000 | 0.1885152 | 0.5692276 |
|      |        |        |           |           |      |       |           |          |

| att3 | MathAtt |        | -1.592717 | 0.3614859 | -4.41 | 0.000 | -2.301216 | -0.8842173 |
|      |        |        |          |           |       |       |          |          |

| att4 | MathAtt |        | -0.8100107 | 0.153064 | -5.29 | 0.000 | -1.11001 | -0.5100108 |
|      |        |        |           |           |       |       |           |          |

| att5 | MathAtt |        | 0.5225423 | 0.1170141 | 4.47 | 0.000 | 0.2931988 | 0.7518858 |
|      |        |        |           |           |      |       |           |          |

| /att1 | cut1 | -1.10264 | 0.1312272 | -1.359741 | -0.8453396 |
|        | cut2 | -0.2495339 | 0.1160385 | -0.478651 | -0.0221027 |
|        | cut3 | 0.2983261 | 0.1164414 | 0.0701052 | 0.5265471 |
|        | cut4 | 1.333053 | 0.1391907 | 1.060244 | 1.605861 |
| /att2 | cut1 | -1.055791 | 0.1062977 | -1.264131 | -0.8474513 |
|        | cut2 | -0.1941211 | 0.0941435 | -0.378639 | -0.096032 |
|        | cut3 | 0.3598488 | 0.0952038 | 0.1732528 | 0.5464448 |
|        | cut4 | 1.132624 | 0.1082204 | 0.920516 | 1.344732 |
| /att3 | cut1 | -1.053519 | 0.1733999 | -1.393377 | -0.7136614 |
|        | cut2 | -0.0491073 | 0.1442846 | -0.3318999 | 0.236853 |
|        | cut3 | 0.5570671 | 0.1538702 | 0.2554871 | 0.8586471 |
|        | cut4 | 1.666859 | 0.2135554 | 1.248298 | 2.08542 |
| /att4 | cut1 | -1.07378 | 0.1214071 | -1.311734 | -0.8358264 |
|        | cut2 | -0.2112462 | 0.1076501 | -0.4222366 | -0.0002559 |
|        | cut3 | 0.406347 | 0.1094847 | 0.191761 | 0.620933 |
|        | cut4 | 1.398185 | 0.1313327 | 1.140778 | 1.655593 |
| /att5 | cut1 | -1.244051 | 0.1148443 | -1.469142 | -1.018961 |
|        | cut2 | -0.336135 | 0.0986678 | -0.5295203 | -0.1427498 |
|        | cut3 | 0.2137776 | 0.0978943 | 0.0219084 | 0.4056468 |
|        | cut4 | 0.9286849 | 0.107172 | 0.7186316 | 1.138738 |

var(MathAb) = 2.300652 0.7479513 4.21 0.000 0.3850796 1.056601
var(MathAtt) = 1.520854 0.4077674 4.01 0.000 0.2931988 0.7518858

cov(MathAb, MathAtt) = 0.8837681 0.2204606 4.01 0.000 0.4516733 1.315863
Notes:

1. The estimated covariance between math attitude and ability is 0.88.

2. There is something new in the output, namely, things labeled cut1, ..., cut4. These appear for each of the five attitudinal measures. These are the ordered logit’s cutpoints, the values on the logit’s distribution that separate attitude 1 from attitude 2, attitude 2 from attitude 3, and so on. The four cutpoints map the continuous distribution into five ordered, categorical groups.

3. There’s something interesting hiding in the MathAtt coefficients: the coefficients for two of the paths, att3 att4 <- MathAtt, are negative! If you look back to the description of the data, you will find that the sense of these two questions was reversed from those of the other questions. Strong agreement on these two questions was agreement with a negative feeling about mathematics.

estat sd displays the fitted variance components as standard deviations and correlations. From the following, we see that the estimated correlation between attitude and ability is 0.4725.

|            | Coef. | Std. Err. | z    | P>|z|   | [95% Conf. Interval] |
|------------|-------|-----------|------|-------|-----------------------|
| sd(MathAb) | 1.51679 | .2465573 | 1.102963 | 2.085883 |
| sd(MathAtt)| 1.233229 | .1653251 | .9482719 | 1.603817 |
| corr(MathAb, MathAtt) | .4724644 | .0649541 | 7.27 | 0.000 | .3357955 .5896506 |

Fitting the model with the Builder

Use the diagram in Fitting the two-factor model above for reference.

1. Open the dataset.
   In the Command window, type
   
   . use https://www.stata-press.com/data/r16/gsem_cfa

2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Put the Builder in gsem mode by clicking on the $\mathbb{G}$ button.

4. Create the measurement component for MathAb.
   Select the Add measurement component tool, $\mathbb{V}$, and then click in the diagram about one-third of the way up from the bottom and slightly left of the center.

In the resulting dialog box,
   a. change the Latent variable name to MathAb;
   b. select q1, q2, q3, q4, q5, q6, q7, and q8 by using the Measurement variables control;
   c. check Make measurements generalized;
   d. select Bernoulli, Logit in the Family/Link control;
   e. select Down in the Measurement direction control;
   f. click on OK.

If you wish, move the component by clicking on any variable and dragging it.
5. Create the measurement component for MathAtt.

Select the Add measurement component tool, 🏡, and then click in the diagram about one-third of the way down from the top and slightly left of the center.

In the resulting dialog box,
   a. change the Latent variable name to MathAtt;
   b. select att1, att2, att3, att4, and att5 by using the Measurement variables control;
   c. check Make measurements generalized;
   d. select Ordinal, Logit in the Family/Link control;
   e. select Up in the Measurement direction control;
   f. click on OK.

If you wish, move the component by clicking on any variable and dragging it.

6. Create the covariance between MathAtt and MathAb.

   a. Select the Add covariance tool, 📈.
   b. Click in the top-left quadrant of the MathAb oval, and drag a covariance to the bottom left of the MathAtt oval.

7. Clean up.

If you do not like where a covariance has been connected to its variable, use the Select tool, 🍁, to simply click on the covariance, and then click on where it connects to an oval and drag the endpoint. You can also change the bow of the covariance by dragging the control point that extends from one end of the selected covariance.

8. Estimate.

Click on the Estimate button, 📊, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder by typing

```
.webgetsem gsem_2fmm
```

Also see

- [SEM] Example 27g — Single-factor measurement model (generalized response)
- [SEM] Example 29g — Two-parameter logistic IRT model
- [SEM] Example 32g — Full structural equation model (generalized response)
- [SEM] Intro 5 — Tour of models
- [SEM] gsem — Generalized structural equation model estimation command
- [SEM] estat sd — Display variance components as standard deviations and correlations
To demonstrate a structural model with a measurement component, we use the same data used in [SEM] Example 31g:

```
use https://www.stata-press.com/data/r16/gsem_cfa
(Fictional math abilities data)
summarize
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>500</td>
<td>10.5</td>
<td>5.772056</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>id</td>
<td>500</td>
<td>50681.71</td>
<td>29081.41</td>
<td>71</td>
<td>100000</td>
</tr>
<tr>
<td>q1</td>
<td>500</td>
<td>.506</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q2</td>
<td>500</td>
<td>.394</td>
<td>.4891242</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q3</td>
<td>500</td>
<td>.534</td>
<td>.4993423</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q4</td>
<td>500</td>
<td>.424</td>
<td>.4946852</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q5</td>
<td>500</td>
<td>.49</td>
<td>.5004006</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q6</td>
<td>500</td>
<td>.434</td>
<td>.4961212</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q7</td>
<td>500</td>
<td>.52</td>
<td>.5001002</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q8</td>
<td>500</td>
<td>.494</td>
<td>.5004647</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>att1</td>
<td>500</td>
<td>2.946</td>
<td>1.607561</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att2</td>
<td>500</td>
<td>2.948</td>
<td>1.561465</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att3</td>
<td>500</td>
<td>2.84</td>
<td>1.640666</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att4</td>
<td>500</td>
<td>2.91</td>
<td>1.566783</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>att5</td>
<td>500</td>
<td>3.086</td>
<td>1.581013</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>test1</td>
<td>500</td>
<td>75.548</td>
<td>5.948653</td>
<td>55</td>
<td>93</td>
</tr>
<tr>
<td>test2</td>
<td>500</td>
<td>80.556</td>
<td>4.976786</td>
<td>65</td>
<td>94</td>
</tr>
<tr>
<td>test3</td>
<td>500</td>
<td>75.572</td>
<td>6.677874</td>
<td>50</td>
<td>94</td>
</tr>
<tr>
<td>test4</td>
<td>500</td>
<td>74.078</td>
<td>8.845587</td>
<td>43</td>
<td>96</td>
</tr>
</tbody>
</table>
```

`_dta:`

1. Fictional data on math ability and attitudes of 500 students from 20 schools.
2. Variables q1-q8 are incorrect/correct (0/1) on individual math questions.
3. Variables att1-att5 are items from a Likert scale measuring each student's attitude toward math.
4. Variables test1-test4 are test scores from tests of four different aspects of mathematical abilities. Range of scores: 0-100.

These data record results from a fictional instrument measuring mathematical ability. Variables q1 through q8 are the items from the instrument.

In this example, we will also be using variables att1 through att5. These are five Likert-scale questions measuring each student’s attitude toward math.

See Structural models 9: Unobserved inputs, outputs, or both in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- Structural model with measurement component
- Fitting the model with the Builder

Structural model with measurement component

We wish to fit the following model:

\[
\begin{align*}
\text{MathAb} & \quad \varepsilon_1 \\
q_1 & \quad \text{Bernoulli} \\
q_2 & \quad \text{Bernoulli} \\
q_3 & \quad \text{Bernoulli} \\
q_4 & \quad \text{Bernoulli} \\
q_5 & \quad \text{Bernoulli} \\
q_6 & \quad \text{Bernoulli} \\
q_7 & \quad \text{Bernoulli} \\
q_8 & \quad \text{Bernoulli}
\end{align*}
\]

\[
\begin{align*}
\text{MathAtt} & \\
\text{att1} & \quad \text{ordinal} \\
\text{att2} & \quad \text{ordinal} \\
\text{att3} & \quad \text{ordinal} \\
\text{att4} & \quad \text{ordinal} \\
\text{att5} & \quad \text{ordinal}
\end{align*}
\]

This is the same model we fit in \[\text{SEM}\] Example 31g, except that rather than a correlation (curved path) between MathAtt and MathAb, this time we assume a direct effect and so allow a straight path.

If you compare the two path diagrams, in addition to the new substitution of the direct path for the curved path signifying correlation, there is now an error variable on MathAb. In the previous diagram, MathAb was exogenous. In this diagram, it is endogenous and thus requires an error term. In the Builder, the error term is added automatically.

To fit this model in the command language, we type

\[
\begin{align*}
. \text{gsem} \ (\text{MathAb} \to q1-q8, \text{logit}) \\
& \quad \text{MathAtt} \to \text{att1-att5, ologit} \\
& \quad \text{MathAtt} \to \text{MathAb}
\end{align*}
\]

Fitting fixed-effects model:
Iteration 0:  log likelihood = -6629.7253
Iteration 1:  log likelihood = -6628.7848
Iteration 2:  log likelihood = -6628.7848
Refining starting values:
Grid node 0:  log likelihood = -6429.1636
Example 32g — Full structural equation model (generalized response)

Fitting full model:
Iteration 0:  log likelihood = -6429.1636
Iteration 1:  log likelihood = -6396.7471
Iteration 2:  log likelihood = -6394.6197
Iteration 3:  log likelihood = -6394.3949
Iteration 4:  log likelihood = -6394.3923
Iteration 5:  log likelihood = -6394.3923

Generalized structural equation model
Number of obs = 500
Response : q1
  Family : Bernoulli
  Link : logit
Response : q2
  Family : Bernoulli
  Link : logit
Response : q3
  Family : Bernoulli
  Link : logit
Response : q4
  Family : Bernoulli
  Link : logit
Response : q5
  Family : Bernoulli
  Link : logit
Response : q6
  Family : Bernoulli
  Link : logit
Response : q7
  Family : Bernoulli
  Link : logit
Response : q8
  Family : Bernoulli
  Link : logit
Response : att1
  Family : ordinal
  Link : logit
Response : att2
  Family : ordinal
  Link : logit
Response : att3
  Family : ordinal
  Link : logit
Response : att4
  Family : ordinal
  Link : logit
Response : att5
  Family : ordinal
  Link : logit
Log likelihood = -6394.3923
( 1) [q1]MathAb = 1
( 2) [att1]MathAtt = 1
|   | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|---|-------|-----------|------|------|----------------------|
| q1 | MathAb | 1 (constrained) | | | |
|   | _cons  | .044612    | .1272967 | 0.35 | 0.726 | -.204885 | .294109 |
| q2 | MathAb | .3446066   | .1050261 | 3.28 | 0.001 | .1387593 | .550454 |
|   | _cons  | -.4572215  | .0979965 | -4.67 | 0.000 | -.6492911 | -.2651519 |
| q3 | MathAb | .5445222   | .1386992 | 3.93 | 0.000 | .2726767 | .8163677 |
|   | _cons  | .1591406   | .1033116 | 1.54 | 0.123 | -.0433465 | .3616276 |
| q4 | MathAb | .2858862   | .0948549 | 3.01 | 0.003 | .099974 | .4717984 |
|   | _cons  | -.3196648  | .0947684 | -3.37 | 0.001 | -.5054075 | -.1339222 |
| q5 | MathAb | .8174769   | .1867022 | 4.38 | 0.000 | .4515473 | 1.183406 |
|   | _cons  | -.04543    | .116575  | -0.39 | 0.697 | -0.2739129 | .1830528 |
| q6 | MathAb | .6030423   | .1471949 | 4.10 | 0.000 | .3145457 | .8915389 |
|   | _cons  | -.3099919  | .1070853 | -2.89 | 0.004 | -.5198754 | -.1001085 |
| q7 | MathAb | .7208369   | .171309  | 4.21 | 0.000 | .3850774 | 1.066597 |
|   | _cons  | .1047264   | .1116494 | 0.94 | 0.348 | -.1141024 | .3235553 |
| q8 | MathAb | .5814736   | .1426725 | 4.08 | 0.000 | .3018406 | .8611067 |
|   | _cons  | -.0250443  | .1045135 | -0.24 | 0.811 | -.2298869 | .1797984 |
| att1 | MathAtt | 1 (constrained) | | | |
| att2 | MathAtt | .3788715   | .0971234 | 3.90 | 0.000 | .1885131 | .5692299 |
| att3 | MathAtt | -1.592717  | .3614956 | -4.41 | 0.000 | -2.301236 | -.8841989 |
| att4 | MathAtt | -.8100108  | .1530675 | -5.29 | 0.000 | -1.110017 | -.510004 |
| att5 | MathAtt | .5225425   | .1170166 | 4.47 | 0.000 | .2931942 | .7518907 |
| MathAb | MathAtt | .581103    | .147776  | 3.93 | 0.000 | .2914987 | .8707072 |
| /att1 | cut1  | -1.10254   | .131228  | | | -1.359742 | -.8453377 |
|      | cut2  | -.2495339  | .1160385 | | | -.4769652 | -.0221025 |
|      | cut3  | .2983261   | .1164415 | | | .070105  | .5265472 |
|      | cut4  | 1.333052   | .1391919 | | | 1.060241 | 1.605864 |
360 Example 32g — Full structural equation model (generalized response)

| /att2 | cut1  | -1.055791 | .1062977 | -1.264131 | -.8474513 |
|       | cut2  | -.1941211 | .0941435 | -.378639  | -.0096032 |
|       | cut3  | .3598488  | .0952038 | .1732528  | .5464448  |
|       | cut4  | 1.132624  | .1082204 | .9205156  | 1.344732  |

| /att3 | cut1  | -1.053519 | .1734001 | -1.393377 | -.7136612 |
|       | cut2  | -.0491074 | .1442846 | -.3319    | .2336853  |
|       | cut3  | .5570672  | .1538702 | .2554871  | .8586472  |
|       | cut4  | 1.666859  | .2135557 | 1.248297  | 2.08542   |

| /att4 | cut1  | -1.07378  | .1214071 | -1.311734 | -.8358264 |
|       | cut2  | -.2112462 | .1076501 | -.4222365 | -.0002559 |
|       | cut3  | .406347   | .1094847 | .191761   | .620933   |
|       | cut4  | 1.398185  | .1313327 | 1.140778  | 1.655593  |

| /att5 | cut1  | -1.244051 | .1148443 | -1.469142 | -1.018961 |
|       | cut2  | -.336135  | .0986678 | -.5295203 | -.1427498 |
|       | cut3  | .2137776  | .0978943 | .0219084  | .4056468  |
|       | cut4  | .9286849  | .107172  | .7186316  | 1.138738  |

| var(e.MathAb) | 1.787117 | .5974753 | .9280606 | 3.441357 |
| var(MathAtt)  | 1.520854 | .4077885 | .8991947 | 2.572298 |

Notes:

1. In the model fit in [SEM] Example 31g, we estimated a correlation between MathAtt and MathAb of 0.4725.

2. Theoretically speaking, the model fit above and the model in [SEM] Example 31g are equivalent. Both posit a linear relationship between the latent variables and merely choose to parameterize the relationship differently. In [SEM] Example 31g, it was parameterized as a covariance. In this example, it is parameterized as causal. People often use structural equation modeling to confirm a proposed hypothesis. It is important that the causal model you specify be based on theory or that you have some other justification. You need something other than empirical results to rule out competing but equivalent models such as the covariance model. Distinguishing causality from correlation is always problematic.

3. Practically speaking, note that the log-likelihood values for this model and the model in [SEM] Example 31g are equal at $-6394.3923$. Also note that the estimated variances of math attitude, var(MathAtt), are also equal at $1.520854$.

Fitting the model with the Builder

Use the diagram in Structural model with measurement component above for reference.

1. Open the dataset.
   In the Command window, type
   
   . use https://www.stata-press.com/data/r16/gsem_cfa

2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.
3. Put the Builder in \texttt{gsem} mode by clicking on the $\mathcal{G}$ button.

4. Create the measurement component for MathAb.
   
   Select the Add Measurement Component tool, $\mathcal{M}$, and then click in the diagram about one-third of the way up from the bottom and slightly left of the center.
   
   In the resulting dialog box,
   
   a. change the \textit{Latent variable name} to MathAb;
   b. select q1, q2, q3, q4, q5, q6, q7, and q8 by using the \textit{Measurement variables} control;
   c. check \textit{Make measurements generalized};
   d. select Bernoulli, Logit in the \textit{Family/Link} control;
   e. select Down in the \textit{Measurement direction} control;
   f. click on \textbf{OK}.
   
   If you wish, move the component by clicking on any variable and dragging it.

5. Create the measurement component for MathAtt.
   
   Select the Add Measurement Component tool, $\mathcal{M}$, and then click in the diagram about one-third of the way down from the top and slightly left of the center.
   
   In the resulting dialog box,
   
   a. change the \textit{Latent variable name} to MathAtt;
   b. select att1, att2, att3, att4, and att5 by using the \textit{Measurement variables} control;
   c. check \textit{Make measurements generalized};
   d. select Ordinal, Logit in the \textit{Family/Link} control;
   e. select Up in the \textit{Measurement direction} control;
   f. click on \textbf{OK}.
   
   If you wish, move the component by clicking on any variable and dragging it.

6. Create path from MathAtt to MathAb.
   
   a. Select the Add Path tool, $\text{−−}$.
   
   b. Click in the bottom of the MathAtt oval (it will highlight when you hover over it), and drag a path to the top of the MathAb oval (it will highlight when you can release to connect the path).

7. Clean up the direction of the error.
   
   The error on MathAb has likely been created below the oval instead of to the right of the oval. If so, choose the Select tool, $\mathcal{H}$, and then click in the MathAb oval. Click on one of the \textbf{Error rotation} buttons, $\leftrightarrow$, in the Contextual Toolbar until the error is where you want it.

8. Estimate.
   
   Click on the \textbf{Estimate} button, $\mathcal{D}$, in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{GSEM estimation options} dialog box.

You can open a completed diagram in the Builder by typing

\texttt{.webgetsem gsem_{sem}}
Also see

[SEM] Example 9 — Structural model with measurement component

[SEM] Example 31g — Two-factor measurement model (generalized response)

[SEM] Intro 5 — Tour of models

[SEM] gsem — Generalized structural equation model estimation command
Example 33g — Logistic regression

Description

In this example, we demonstrate with `gsem` how to fit a standard logistic regression, which is often referred to as the logit model in generalized linear model (GLM) framework.

```
use https://www.stata-press.com/data/r16/gsem_lbw
(Hosmer & Lemeshow data)

. describe
Contains data from https://www.stata-press.com/data/r16/gsem_lbw.dta
obs: 189  Hosmer & Lemeshow data
vars: 11   21 Mar 2018 12:28
(_dta has notes)

storage  display value
variable name type format label
id        int   %8.0g   subject id
low       byte  %8.0g   birth weight < 2500g
age       byte  %8.0g   age of mother
lwt       int   %8.0g   weight, last menstrual period
race      byte  %8.0g   race
smoke     byte  %9.0g   smoke smoked during pregnancy
ptl       byte  %8.0g   premature labor history (count)
h         byte  %8.0g   has history of hypertension
ui        byte  %8.0g   presence, uterine irritability
ftv       byte  %8.0g   # physician visits, 1st trimester
bwt       int   %8.0g   birth weight (g)
```

Sorted by:
```
. notes
_dta:
2. Data from a study of risk factors associated with low birth weights.

See Structural models 3: Binary-outcome models in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the logit model
- Obtaining odds ratios
- Fitting the model with the Builder

363
Fitting the logit model

The model we wish to fit is

That is, we wish to fit a model in which low birthweight is determined by a history of hypertension (ht), mother’s age (age), mother’s weight at last menstrual period (lwt), mother’s race (white, black, or other; race), whether the mother smoked during pregnancy (smoke), the number of premature babies previously born to the mother (ptl), and whether the mother has suffered from the presence of uterine irritability (ui).

The path diagram matches the variable names listed in parentheses above except for race, where the path diagram contains not one but three boxes filled in with 1b.race, 2.race, and 3.race. This is because in our dataset, race is coded 1, 2, or 3, meaning white, black, or other. We want to include indicator variables for race so that we have a separate coefficient for each race. Thus we need three boxes.

In Stata, 1.race means “an indicator for race equaling 1”. Thus it should not surprise you if you filled in the boxes with 1.race, 2.race, and 3.race, and that is almost what we did. The difference is that we filled in the first box with 1b.race rather than 1.race. We use the b to specify the base category, which we specified as white. If we wanted the base category to be black, we would have specified 2b.race and left 1.race alone.
The above is called factor-variable notation. See [SEM] Intro 3 for details on using factor-variable notation with gsem.

In the command language, we could type

```
.gsem (low <- age lwt 1b.race 2.race 3.race smoke ptl ht ui), logit
```

to fit the model. Written that way, there is a one-to-one correspondence to what we would type and what we would draw in the Builder. The command language, however, has a feature that will allow us to type `i.race` instead of `1b.race 2.race 3.race`. To fit the model, we could type

```
.gsem (low <- age lwt i.race smoke ptl ht ui), logit
```

`i.varname` is a command-language shorthand for specifying indicators for all the levels of a variable and using the first level as the base category. You can use `i.varname` in the command language but not in path diagrams because boxes can contain only one variable. In the Builder, however, you will discover a neat feature so that you can type `i.race` and the Builder will create however many boxes are needed for you, filled in, with the first category marked as the base. We will explain below how you do that.

The result of typing our estimation command is

```
.gsem (low <- age lwt i.race smoke ptl ht ui), logit
Iteration 0: log likelihood = -101.0213
Iteration 1: log likelihood = -100.72519
Iteration 2: log likelihood = -100.724
Iteration 3: log likelihood = -100.724
Generalized structural equation model Number of obs = 189
Response : low
Family : Bernoulli
Link : logit
Log likelihood = -100.724

|             | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------------|--------|-----------|-------|------|---------------------|
| low         |        |           |       |      |                     |
| age         | -.0271003 | .0364504 | -0.74 | 0.457 | -.0985418 to .0443412 |
| lwt         | -.0151508 | .0069259 | -2.19 | 0.029 | -.0287253 to -.0015763 |
| race        |         |           |       |      |                     |
| black       | 1.262647 | .5264101 | 2.40  | 0.016 | .2309024 to 2.294392 |
| other       | .8620792 | .4391532 | 1.96  | 0.050 | .0013548 to 1.722804 |
| smoke       | .9233448 | .4008266 | 2.30  | 0.021 | .137739 to 1.708951 |
| ptl         | .5418366 | .346249  | 1.56  | 0.118 | -.136799 to 1.220472 |
| ht          | 1.832518 | .6916292 | 2.65  | 0.008 | .4769494 to 3.188086 |
| ui          | .7585135 | .4593768 | 1.65  | 0.099 | -.1418484 to 1.658875 |
| _cons       | .4612239 | 1.20459  | 0.38  | 0.702 | -1.899729 to 2.822176 |
```

Obtaining odds ratios

Some of you are looking at the output above, nodding your heads, and thinking to yourselves, “Yes, that’s right.” Others are shaking your heads sadly and thinking, “Where are the exponentiated coefficients, the odds ratios?” Researchers from different backgrounds are used to seeing logit results presented in two different ways.
If you want to see the odds ratios, type `estat eform` after fitting the model:

```
. estat eform
```

|       | exp(b)   | Std. Err. |    z  | P>|z|  | [95% Conf. Interval] |
|-------|----------|-----------|-------|------|---------------------|
| low   |          |           |       |      |                     |
| age   | .9732636 | .0354759  | -0.74 | 0.457| .9061578 1.045339   |
| lwt   | .9849634 | .0068217  | -2.19 | 0.029| .9716834 .9984249   |
|       |          |           |       |      |                     |
| race  |          |           |       |      |                     |
| black | 3.534767 | 1.860737  | 2.40  | 0.016| 1.259736 9.918406   |
| other | 2.368079 | 1.039949  | 1.96  | 0.050| 1.001356 5.600207   |
|       |          |           |       |      |                     |
| smoke |          |           |       |      |                     |
| ptl   | 2.517698 | 1.00916   | 2.30  | 0.021| 1.147676 5.523162   |
| ht    | 1.719161 | .5952579  | 1.56  | 0.118| .8721455 3.388787   |
| ui    | 6.249602 | 4.322408  | 2.65  | 0.008| 1.611152 24.24199   |
| _cons | 2.1351   | .9808153  | 1.65  | 0.099| .8677528 5.2534     |

Whichever way you look at the results above, they are identical to the results that would be produced by typing

```
. logit low age lwt i.race smoke ptl ht ui
```

or

```
. logistic low age lwt i.race smoke ptl ht ui
```

which are two other ways that Stata can fit logit models. `logit`, like `gsem`, reports coefficients by default. `logistic` reports odds ratios by default.

### Fitting the model with the Builder

Use the diagram in *Fitting the logit model* above for reference.

1. Open the dataset.

   In the Command window, type
   
   ```
   . use https://www.stata-press.com/data/r16/gsem_lbw
   ```

2. Open a new Builder diagram.

   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Put the Builder in gsem mode by clicking on the button.

4. Enlarge the size of the canvas to accommodate the height of the diagram.

   Click on the Adjust canvas size button, in the Standard Toolbar, change the second size to 5 (inches), and then click on OK.

5. Create the logistic regression component for `low`.

   Select the Add regression component tool, and then click in the diagram about one-third of the way in from the left and halfway down.

   In the resulting dialog box,
   
   a. select `low` in the Dependent variable control;
Example 33g — Logistic regression

b. check Make measurements generalized;
c. select Bernoulli, Logit in the Family/Link control;
d. select the Select variables radio button (it may already be selected);
e. use the Independent variables control to select the variables age and lwt;
f. include the levels of the factor variable race by clicking on the "" button next to the Independent variables control. In the resulting dialog box, select the Factor variable radio button, select Main effect in the Specification control, and select race in the Variables control for Variable 1. Click on Add to varlist, and then click on OK;
g. continue with the Independent variables control to select the variables smoke, ptl, ht, and ui;
h. select Left in the Independent variables’ direction control;
i. click on OK.

If you wish, move the component by clicking on any variable and dragging it.

6. Clean up.
   The box for low is created closer to the independent variables than it is in the example diagram. Use the Select tool, □, and click on the box for low. Drag it to the right to allow more space for results along the paths.

7. Estimate.
   Click on the Estimate button, ▶, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder by typing

   . webgetsem gsem_logit

Reference


Also see

[SEM] Example 34g — Combined models (generalized responses)
[SEM] Example 35g — Ordered probit and ordered logit
[SEM] Example 37g — Multinomial logistic regression
[SEM] Intro 3 — Learning the language: Factor-variable notation (gsem only)
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
[SEM] estat eform — Display exponentiated coefficients
We demonstrate how to fit a combined model with one Poisson regression and one logit regression by using the following data:

```
. use https://www.stata-press.com/data/r16/gsem_lbw
(Hosmer & Lemeshow data)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_lbw.dta
    obs: 189  Hosmer & Lemeshow data
    vars: 11  21 Mar 2018 12:28
 (_dta has notes)

storage  display value
variable name type format label variable label
id        int %8.0g label    subject id
low       byte %8.0g label   birth weight < 2500g
age       byte %8.0g label   age of mother
lwt       int %8.0g label   weight, last menstrual period
race      byte %8.0g label   race
smoke     byte %9.0g label   smoked during pregnancy
ptl       byte %8.0g label   premature labor history (count)
h        byte %8.0g label   has history of hypertension
ui        byte %8.0g label   presence, uterine irritability
ftv       byte %8.0g label   # physician visits, 1st trimester
bwt       int %8.0g label   birth weight (g)
```

Sorted by:
```
. notes
_dta:
  2. Data from a study of risk factors associated with low birth weights.

See Structural models 8: Dependencies between response variables in [SEM] Intro 5 for background.
Fitting the combined model

The model we wish to fit is

This model has one logit equation and one Poisson regression equation, with the Poisson response variable also being an explanatory variable in the logit equation.

Because the two equations are recursive, it is not necessary to fit these models together. We could draw separate diagrams for each equation and fit each separately. Even so, many researchers often do fit recursive models together, and sometimes, it is just the first step before placing constraints across models or introducing a common latent variable. The latter might be likely in this case because neither generalized linear response has an error that could be correlated and so the only way to correlate these two responses in \texttt{gsem} is to add a shared latent variable affecting each.

Our purpose here is to show that you can mix models with generalized response variables of different types.
To fit the model in the command language, we type

```
. gsem (low <- ptl age smoke ht lwt i.race ui, logit)
> (ptl <- age smoke ht, poisson)
```

Iteration 0: log likelihood = -322.96738  
Iteration 1: log likelihood = -200.5818  
Iteration 2: log likelihood = -198.58086  
Iteration 3: log likelihood = -198.56179  
Iteration 4: log likelihood = -198.56178

Generalized structural equation model

Number of obs = 189
Response : low
Family : Bernoulli
Link : logit
Response : ptl
Family : Poisson
Link : log

Log likelihood = -198.56178

|     | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-----|--------|-----------|-------|------|---------------------|
| low |        |           |       |      |                     |                     |
| ptl | .5418366 | .346249 | 1.56  | 0.118| -.136799            | 1.220472            |
| age | -.0271003 | .0364504 | -0.74 | 0.457| -.0985418           | .0443412            |
| smoke | .9233448 | .4008266 | 2.30  | 0.021| .137739             | 1.708951            |
| ht   | 1.832518 | .6916292 | 2.65  | 0.008| .4769494            | 3.188086            |
| lwt  | -.0151508 | .0069259 | -2.19 | 0.029| -.0287253           | -.0015763           |
| race |        |           |       |      |                     |                     |
| black | 1.262647 | .5264101 | 2.40  | 0.016| .2309023            | 2.294392            |
| other | .8620791 | .4391532 | 1.96  | 0.050| .0013548            | 1.722803            |
| ui   | .7585135 | .4593768 | 1.65  | 0.099| -.1418484           | 1.658875            |
| _cons | .4612238 | 1.20459 | 0.38  | 0.702| -1.899729           | 2.822176            |
| ptl  |        |           |       |      |                     |                     |
| age  | .0370598 | .0298752 | 1.24  | 0.215| -.0214946           | .0956142            |
| smoke | .9602534 | .3395867 | 2.83  | 0.005| .2944796            | 1.626027            |
| ht   | -.1853501 | .7271851 | -0.25 | 0.799| -1.610607           | 1.239906            |
| _cons | -2.985512 | .7842174 | -3.81 | 0.000| -4.52255            | -1.448474           |

Obtaining odds ratios and incidence-rate ratios

As mentioned in [SEM] Example 33g, some researchers prefer to see exponentiated coefficients. In both odds ratios and incidence-rate ratios, exponentiation is meaningful. Exponentiated logit coefficients are odds ratios, and exponentiated Poisson regression coefficients are incidence-rate ratios. To obtain exponentiated coefficients for both equations, we type
### Example 34g — Combined models (generalized responses)

```
. estat eform low pti

|          | exp(b)   | Std. Err. |     z  |   P>|z| |   [95% Conf. Interval] |
|----------|----------|-----------|--------|--------|------------------------|
| low pti  | 1.719161 | .5952579  | 1.56   | 0.118  | .8721455 - 3.388787    |
| age      | .9732636 | .0354759  | -0.74  | 0.457  | .9061578 - 1.045339    |
| smoke    | 2.517698 | 1.00916   | 2.30   | 0.021  | 1.147676 - 5.523162    |
| ht       | 6.249602 | 4.322407  | 2.65   | 0.008  | 1.611152 - 24.24199    |
| lwt      | .9849634 | .0068217  | -2.19  | 0.029  | .9716834 - .9984249    |
| race     |          |           |        |        |                        |
| black    | 3.534767 | 1.860737  | 2.40   | 0.016  | 1.259736 - 9.918406    |
| other    | 2.368079 | 1.039949  | 1.96   | 0.050  | 1.001356 - 5.600207    |
| ui       | 2.1351   | .9808153  | 1.65   | 0.099  | .8677528 - 5.2534      |
| _cons    | 1.586014 | 1.910496  | 0.38   | 0.702  | .1496092 - 16.8134     |
| pti      |          |           |        |        |                        |
| age      | 1.037755 | .0310032  | 1.24   | 0.215  | .9787348 - 1.100334    |
| smoke    | 2.612358 | .8873835  | 2.83   | 0.005  | 1.342428 - 5.083638    |
| ht       | .8308134 | .6041551  | -0.25  | 0.799  | .1997664 - 3.45529     |
| _cons    | .0505137 | .0396137  | -3.81  | 0.000  | .0108613 - .2349286    |
```

Had we merely typed `estat eform` without the two equation names, we would have obtained exponentiated coefficients for the first equation only.

Equation names are easily found on the output or the path diagrams. Equations are named after the dependent variable.

### Fitting the model with the Builder

Use the diagram in *Fitting the combined model* above for reference.

1. Open the dataset.
   
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/gsem_lbw
   ```

2. Open a new Builder diagram.
   
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in `gsem` mode by clicking on the $\mathcal{G}$ button.

4. Create the independent variables.
   
   Select the Add observed variables set tool, $\mathcal{G}$, and then click at the bottom of the diagram about one-third of the way in from the left.

   In the resulting dialog box,
   
   a. select the **Select variables** radio button (it may already be selected);
   b. use the **Variables** control to select the variables `age`, `smoke`, `ht`, and `lwt` in this order;
   c. include the levels of the factor variable `race` by clicking on the $\mathcal{G}$ button next to the **Variables** control. In the resulting dialog box, select the **Factor variable** radio button, select `Main effect` in the **Specification** control, and select `race` in the **Variables** control for `Variable 1`. Click on **Add to varlist**, and then click on **OK**;
d. continue with the Variables control and select the variable *ui*;
e. select *Vertical* in the Orientation control;
f. click on OK.

If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the generalized response for premature labor history.
   a. Select the Add generalized response variable tool, □.
   b. Click about one-third of the way in from the right side of the diagram, to the right of *ht*.
   c. In the Contextual Toolbar, select Poisson, Log in the Family/Link control.
   d. In the Contextual Toolbar, select *ptl* in the Variable control.

   a. Select the Add generalized response variable tool, □.
   b. Click about one-third of the way in from the right side of the diagram, to the right of *2.race*.
   c. In the Contextual Toolbar, select Bernoulli, Logit in the Family/Link control.
   d. In the Contextual Toolbar, select *low* in the Variable control.

7. Create paths from the independent variables to the dependent variables.
   a. Select the Add path tool, −.
   b. Click in the right side of the *age* rectangle (it will highlight when you hover over it), and drag a path to the left side of the *ptl* rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the − tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable:

   
   ```
   smoke -> ptl  
   ht -> ptl  
   age -> low  
   smoke -> low  
   ht -> low  
   lwt -> low  
   1b.race -> low  
   2.race -> low  
   3.race -> low  
   ui -> low  
   ```
   d. Continuing with the − tool, create the path from *ptl* to *low* by clicking in the bottom of the *ptl* rectangle and dragging the path to the top of the *low* rectangle.
8. Clean up.

If you do not like where a path has been connected to its variables, use the Select tool, \( \text{Select} \), to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.


Click on the **Estimate** button, \( \text{Estimate} \), in the Standard Toolbar, and then click on **OK** in the resulting *GSEM estimation options* dialog box.

You can open a completed diagram in the Builder by typing

\[ \text{. webgetsem gsem_comb} \]

**Reference**


**Also see**

[SEM] Example 33g — Logistic regression

[SEM] Example 45g — Heckman selection model

[SEM] Example 46g — Endogenous treatment-effects model

[SEM] Intro 5 — Tour of models

[SEM] gsem — Generalized structural equation model estimation command

[SEM] estat eform — Display exponentiated coefficients
**Description**

Below we demonstrate ordered probit and ordered logit in a measurement-model context. We are not going to illustrate every family/link combination. Ordered probit and logit, however, are unique in that a single equation is able to predict a set of ordered outcomes. The unordered alternative, mlogit, requires $k - 1$ equations to fit $k$ (unordered) outcomes.

To demonstrate ordered probit and ordered logit, we use the following data:

```stata
use https://www.stata-press.com/data/r16/gsem_issp93
(description)
```

Contains data from https://www.stata-press.com/data/r16/gsem_issp93.dta

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>%9.0g</td>
<td></td>
<td>respondent identifier</td>
</tr>
<tr>
<td>y1</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>too much science, not enough feelings &amp; faith</td>
</tr>
<tr>
<td>y2</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>science does more harm than good</td>
</tr>
<tr>
<td>y3</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>any change makes nature worse</td>
</tr>
<tr>
<td>y4</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>science will solve environmental problems</td>
</tr>
<tr>
<td>sex</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>sex</td>
</tr>
<tr>
<td>age</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>age (6 categories)</td>
</tr>
<tr>
<td>edu</td>
<td>byte</td>
<td>%20.0g</td>
<td></td>
<td>education (6 categories)</td>
</tr>
</tbody>
</table>

Sorted by:

notes

\_dta:

2. Full text of y1: We believe too often in science, and not enough in feelings and faith.
3. Full text of y2: Overall, modern science does more harm than good.
4. Full text of y3: Any change humans cause in nature, no matter how scientific, is likely to make things worse.
5. Full text of y4: Modern science will solve our environmental problems with little change to our way of life.

See *Structural models 5: Ordinal models* in [SEM] Intro 5 for background.
Remarks and examples

Remarks are presented under the following headings:

- Ordered probit
- Ordered logit
- Fitting the model with the Builder

Ordered probit

For the measurement model, we focus on variables $y_1$ through $y_4$. Each variable contains 1–5, with 1 meaning strong disagreement and 5 meaning strong agreement with a statement about science.

Ordered probit produces predictions about the probabilities that a respondent gives response 1, response 2, ..., response $k$. It does this by dividing up the domain of an $N(0, 1)$ distribution into $k$ categories defined by $k - 1$ cutpoints, $c_1, c_2, \ldots, c_{k-1}$. Individual respondents are assumed to have a score $s = X\beta + \epsilon$, where $\epsilon \sim N(0, 1)$, and then that score is used along with the cutpoints to produce probabilities for each respondent producing response 1, 2, ..., $k$.

$$\Pr(\text{response is } i \mid X) = \Pr(c_{i-1} < X\beta + \epsilon \leq c_i)$$

where $c_0 = -\infty$; $c_k = +\infty$; and $c_1, c_2, \ldots, c_{k-1}$ and $\beta$ are parameters of the model to be fit. This ordered probit model has long been known in Stata circles as oprobit.

We have a set of four questions designed to determine the respondent’s attitude toward science, each question with $k = 5$ possible answers ranging on a Likert scale from 1 to 5. With ordered probit in hand, we have a way to take a continuous variable, say, a latent variable we will call $\text{SciAtt}$, and produce predicted categorical responses.

The measurement model we want to fit is

![Diagram showing the measurement model with SciAtt as the latent variable and y1, y2, y3, y4 as the observed variables.]}
We fit the model in the command language by typing

\[ . \text{gsem (y1 y2 y3 y4 <- SciAtt), oprobit} \]

Fitting fixed-effects model:
Iteration 0: log likelihood = -5227.8743
Iteration 1: log likelihood = -5227.8743

Refining starting values:
Grid node 0: log likelihood = -5230.8106

Fitting full model:
Iteration 0: log likelihood = -5230.8106 (not concave)
Iteration 1: log likelihood = -5132.1849 (not concave)
Iteration 2: log likelihood = -5069.5037
Iteration 3: log likelihood = -5040.4779
Iteration 4: log likelihood = -5040.2397
Iteration 5: log likelihood = -5039.8242
Iteration 6: log likelihood = -5039.823
Iteration 7: log likelihood = -5039.823

Generalized structural equation model

| Response | Family | Link | Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|----------|--------|------|-------|-----------|-------|-------|----------------------|
| y1       | ordinal| probit | SciAtt | 1         | 1 (constrained) |
| y2       | ordinal| probit | SciAtt | 1.424366  | .2126574 | 6.70  | 0.000                | 1.007565 - 1.841167 |
| y3       | ordinal| probit | SciAtt | 1.283359  | .1797557 | 7.14  | 0.000                | .931044 - 1.635674   |
| y4       | ordinal| probit | SciAtt | -.0322354 | .0612282 | -0.53 | 0.599                | -.1522405 - .0877697 |

\( /y1 \) [y1]SciAtt = 1

| cut1     | -1.343148 | .0726927 | -1.485623 | -1.200673 |
| cut2     | .0084719  | .0521512 | -.0937426 | .1106863  |
| cut3     | .7876538  | .0595266 | .6709837  | .9043238  |
| cut4     | 1.989985  | .0999181 | 1.794149  | 2.18582   |
Example 35g — Ordered probit and ordered logit

<table>
<thead>
<tr>
<th>/y2</th>
<th>cut1</th>
<th>cut2</th>
<th>cut3</th>
<th>cut4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-1.997245</td>
<td>.1311972</td>
<td>-2.254387</td>
<td>-1.740104</td>
</tr>
<tr>
<td></td>
<td>-.8240241</td>
<td>.0753839</td>
<td>-.9717738</td>
<td>-.6762743</td>
</tr>
<tr>
<td></td>
<td>.0547025</td>
<td>.0606036</td>
<td>-.0640784</td>
<td>.1734834</td>
</tr>
<tr>
<td></td>
<td>1.419923</td>
<td>.1001258</td>
<td>1.22368</td>
<td>1.616166</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>/y3</th>
<th>cut1</th>
<th>cut2</th>
<th>cut3</th>
<th>cut4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-1.271915</td>
<td>.0847483</td>
<td>-1.438019</td>
<td>-1.105812</td>
</tr>
<tr>
<td></td>
<td>.1249493</td>
<td>.0579103</td>
<td>.0114472</td>
<td>.2384515</td>
</tr>
<tr>
<td></td>
<td>.9752553</td>
<td>.0745052</td>
<td>.8292277</td>
<td>1.121283</td>
</tr>
<tr>
<td></td>
<td>2.130661</td>
<td>.1257447</td>
<td>1.884206</td>
<td>2.377116</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>/y4</th>
<th>cut1</th>
<th>cut2</th>
<th>cut3</th>
<th>cut4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-1.484063</td>
<td>.0646856</td>
<td>-1.610844</td>
<td>-1.357281</td>
</tr>
<tr>
<td></td>
<td>-.4259356</td>
<td>.0439145</td>
<td>-.5120065</td>
<td>-.3398647</td>
</tr>
<tr>
<td></td>
<td>.1688777</td>
<td>.0427052</td>
<td>.0851771</td>
<td>.2525782</td>
</tr>
<tr>
<td></td>
<td>.9413113</td>
<td>.0500906</td>
<td>.8431356</td>
<td>1.039487</td>
</tr>
</tbody>
</table>

| var(SciAtt) | .5265523 | .0979611 | .3656637 | .7582305 |

Notes:

1. The cutpoints $c_1, \ldots, c_4$ are labeled cut1, \ldots, cut4 in the output. We have a separate cutpoint for each of the four questions $y_1, \ldots, y_4$. Look at the estimated cutpoints for $y_1$, which are $-1.343, 0.008, 0.788,$ and $1.99$. The probabilities that a person with $\text{SciAtt} = 0$ (its mean) would give the various responses are

   \[
   \begin{align*}
   \Pr(\text{response 1}) &= \text{normal}(-1.343) = 0.090 \\
   \Pr(\text{response 2}) &= \text{normal}(0.008) - \text{normal}(-1.343) = 0.414 \\
   \Pr(\text{response 3}) &= \text{normal}(0.788) - \text{normal}(0.008) = 0.281 \\
   \Pr(\text{response 4}) &= \text{normal}(1.99) - \text{normal}(0.788) = 0.192 \\
   \Pr(\text{response 5}) &= 1 - \text{normal}(1.99) = 0.023
   \end{align*}
   \]

2. The path coefficients ($y_1 \ y_2 \ y_3 \ y_4 \leftarrow \text{SciAtt}$) measure the effect of the latent variable we called science attitude on each of the responses.

3. The estimated path coefficients are 1, 1.42, 1.28, and $-0.03$ for the four questions.

4. If you read the questions—they are listed above—you will find that in all but the fourth question, agreement signifies a negative attitude toward science. Thus $\text{SciAtt}$ measures a negative attitude toward science because the loadings on negative questions are positive and the loading on the single positive question is negative.

5. The direction of the meanings of latent variables is always a priori indeterminate and is set by the identifying restrictions we apply. We applied—or more correctly, \texttt{gsem} applied for us—the constraint that $y_1 \leftarrow \text{SciAtt}$ has path coefficient 1. Because statement 1 was a negative statement about science, that was sufficient to set the direction of $\text{SciAtt}$ to be the opposite of what we hoped for.

   The direction does not matter. You simply must remember to interpret the latent variable correctly when reading results based on it. In the models we fit, including more complicated models, the signs of the coefficients will work themselves out to adjust for the direction of the variable.
Ordered logit

The description of the ordered logit model is identical to that of the ordered probit model except that where we assumed a normal distribution in our explanation above, we now assume a logit distribution. The distributions are similar.

To fit an ordered logit (ologit) model, the link function shown in the boxes merely changes from probit to logit:

```
SciAtt
  y1
  y2
  y3
  y4
```

We can fit the model in the command language by typing

```
. gsem (y1 y2 y3 y4 <- SciAtt), ologit
```

```
Fitting fixed-effects model:
 Iteration 0:  log likelihood = -5227.8743
 Iteration 1:  log likelihood = -5227.8743
```

```
Refining starting values:
 Grid node 0:   log likelihood = -5127.9026
```

```
Fitting full model:
 Iteration 0:  log likelihood = -5127.9026 (not concave)
 Iteration 1:  log likelihood = -5065.4679
 Iteration 2:  log likelihood = -5035.9766
 Iteration 3:  log likelihood = -5035.0943
 Iteration 4:  log likelihood = -5035.0353
 Iteration 5:  log likelihood = -5035.0352
```

```
Generalized structural equation model  Number of obs  =  871
 Response  :  y1
  Family    :  ordinal
     Link    :  logit
 Response  :  y2
  Family    :  ordinal
     Link    :  logit
 Response  :  y3
  Family    :  ordinal
     Link    :  logit
 Response  :  y4
  Family    :  ordinal
     Link    :  logit
Log likelihood = -5035.0352
( 1)  [y1]SciAtt = 1
```
### Example 35g — Ordered probit and ordered logit

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|-------|----------------------|
| **y1** | SciAtt 1 (constrained) | | | |
| **y2** | SciAtt | | | | |
| 1.394767 | .2065479 | 6.75 | 0.000 | .9899406 | 1.799593 |
| **y3** | SciAtt | | | | |
| 1.29383 | .1845113 | 7.01 | 0.000 | .9321939 | 1.655465 |
| **y4** | SciAtt | | | | |
| -.0412446 | .0619936 | -0.67 | 0.506 | -.1627498 | .0802606 |
| /y1 | cut1 | | | | |
| -2.38274 | .1394292 | | | | |
| cut2 | -.0088393 | .0819718 | | | |
| cut3 | 1.326292 | .106275 | | | |
| cut4 | 3.522017 | .195535 | | | |
| /y2 | cut1 | | | | |
| -3.514171 | .2426595 | | | | |
| cut2 | -1.421711 | .135695 | | | |
| cut3 | .0963154 | .1046839 | | | |
| cut4 | 2.491459 | .185535 | | | |
| /y3 | cut1 | | | | |
| -2.263557 | .1618806 | | | | |
| cut2 | .2024798 | .1012122 | | | |
| cut3 | 1.695997 | .139366 | | | |
| cut4 | 3.828154 | .2464566 | | | |
| /y4 | cut1 | | | | |
| -2.606013 | .1338801 | | | | |
| cut2 | -.6866199 | .0718998 | | | |
| cut3 | .268862 | .0684577 | | | |
| cut4 | 1.561921 | .0895438 | | | |
| var(SciAtt) | | | | | |
| 1.715641 | .3207998 | | | | |

Note:
1. Results are nearly identical to those reported for ordered probit.

### Fitting the model with the Builder

Use the diagram in *Ordered probit* above for reference.

1. Open the dataset.
   In the Command window, type
   ```
   use https://www.stata-press.com/data/r16/gsem_issp93
   ```

2. Open a new Builder diagram.
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation*.

3. Put the Builder in *gsem* mode by clicking on the *g* button.
4. Create the measurement component for SciAtt.

Select the Add measurement component tool, \( \text{Add} \), and then click in the diagram about one-third of the way down from the top and slightly left of the center.

In the resulting dialog box,
   a. change the Latent variable name to SciAtt;
   b. select y1, y2, y3, and y4 by using the Measurement variables control;
   c. check Make measurements generalized;
   d. select Ordinal, Probit in the Family/Link control;
   e. select Down in the Measurement direction control;
   f. click on OK.

If you wish, move the component by clicking on any variable and dragging it.

5. Estimate.

Click on the Estimate button, \( \text{Estimate} \), in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

6. To fit the model in Ordered logit, change the type of generalized response for each of the measurement variables.

   a. Choose the Select tool, \( \text{Select} \).
   b. Click on the y1 rectangle. In the Contextual Toolbar, select Ordinal, Logit in the Family/Link control.
   c. Repeat this process to change the family and link to Ordinal, Logit for y2, y3, and y4.

7. Estimate again.

Click on the Estimate button, \( \text{Estimate} \), in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram for the ordered probit model in the Builder by typing

```
.webgetsem gsem_oprobit
```

You can open a completed diagram for the ordered logit model in the Builder by typing

```
.webgetsem gsem_ologit
```

**Reference**

Also see

[SEM] **Example 1** — Single-factor measurement model
[SEM] **Example 27g** — Single-factor measurement model (generalized response)
[SEM] **Example 33g** — Logistic regression
[SEM] **Example 36g** — MIMIC model (generalized response)
[SEM] **Example 37g** — Multinomial logistic regression
[SEM] **Intro 5** — Tour of models
[SEM] **gsem** — Generalized structural equation model estimation command
Example 36g — MIMIC model (generalized response)

Description

To demonstrate a multiple-indicators multiple-causes (MIMIC) model with generalized indicators, we use the same data used in [SEM] Example 35g:

```
. use https://www.stata-press.com/data/r16/gsem_issp93
     (Selection from ISSP 1993)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_issp93.dta
     obs: 871 Selection for ISSP 1993
     vars: 8 21 Mar 2018 16:03
          (_dta has notes)
```

<table>
<thead>
<tr>
<th>variable name</th>
<th>type</th>
<th>format</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>%9.0g</td>
<td></td>
<td>respondent identifier</td>
</tr>
<tr>
<td>y1</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>too much science, not enough feelings &amp; faith</td>
</tr>
<tr>
<td>y2</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>science does more harm than good</td>
</tr>
<tr>
<td>y3</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>any change makes nature worse</td>
</tr>
<tr>
<td>y4</td>
<td>byte</td>
<td>%26.0g</td>
<td>agree5</td>
<td>science will solve environmental problems</td>
</tr>
<tr>
<td>sex</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>sex</td>
</tr>
<tr>
<td>age</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>age (6 categories)</td>
</tr>
<tr>
<td>edu</td>
<td>byte</td>
<td>%20.0g</td>
<td></td>
<td>education (6 categories)</td>
</tr>
</tbody>
</table>

Sorted by:
. notes
_dta:
2. Full text of y1: We believe too often in science, and not enough in feelings and faith.
3. Full text of y2: Overall, modern science does more harm than good.
4. Full text of y3: Any change humans cause in nature, no matter how scientific, is likely to make things worse.
5. Full text of y4: Modern science will solve our environmental problems with little change to our way of life.

See _Structural models 10: MIMIC models_ in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the MIMIC model
- Fitting the model with the Builder
Fitting the MIMIC model

For a refresher on MIMIC models, see [SEM] Example 10. In the data above, we have ordered categorical indicators. For an explanation on how we are about to treat these indicators with ordered probit, see [SEM] Example 35g.

We wish to fit the following model:

\[
\begin{align*}
\text{SciAtt} & \leftrightarrow \varepsilon_1 \\
\text{sex} & \rightarrow \text{SciAtt} \\
\text{y}_1 & \rightarrow \text{SciAtt} \\
\text{y}_2 & \rightarrow \text{SciAtt} \\
\text{y}_3 & \rightarrow \text{SciAtt} \\
\text{y}_4 & \rightarrow \text{SciAtt}
\end{align*}
\]

We fit the model in the command language by typing

```stata
.gsem (y1 y2 y3 y4 <- SciAtt) (SciAtt <- sex), oprobit
```

Fitting fixed-effects model:
Iteration 0:  log likelihood = -5227.8743
Iteration 1:  log likelihood = -5227.8743

Refining starting values:
Grid node 0:  log likelihood = -5230.8106

Fitting full model:
Iteration 0:  log likelihood = -5230.8106  (not concave)
Iteration 1:  log likelihood = -5230.8106  (not concave)
Iteration 2:  log likelihood = -5066.8481
Iteration 3:  log likelihood = -5033.2807
Iteration 4:  log likelihood = -5032.7884
Iteration 5:  log likelihood = -5032.3837
Iteration 6:  log likelihood = -5032.3778
Iteration 7:  log likelihood = -5032.3778
### Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>Family</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>y1</td>
<td>ordinal</td>
<td>probit</td>
</tr>
<tr>
<td>y2</td>
<td>ordinal</td>
<td>probit</td>
</tr>
<tr>
<td>y3</td>
<td>ordinal</td>
<td>probit</td>
</tr>
<tr>
<td>y4</td>
<td>ordinal</td>
<td>probit</td>
</tr>
</tbody>
</table>

Number of obs = 871  
Log likelihood = -5032.3778

( 1) [y1]SciAtt = 1

|   | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|---|-------|-----------|------|------|---------------------|
| y1 SciAtt | 1 (constrained) |
| y2 SciAtt | 1.405732 | 0.2089672 | 6.73 | 0.000 | 0.9961641 - 1.8153 |
| y3 SciAtt | 1.246449 | 0.1710771 | 7.29 | 0.000 | 0.911144 - 1.581754 |
| y4 SciAtt | 0.0345517 | 0.0602017 | -0.57 | 0.566 | -0.0834415 0.1525494 |
| SciAtt sex | -0.2337427 | 0.0644245 | -3.63 | 0.000 | -0.3600124 - 0.1074729 |
| /y1 cut1 | -1.469615 | 0.0855651 | 1.63732 | -1.301911 |
| cut2 | -0.10992 | 0.0615897 | 1.63732 | -0.107937 |
| cut3 | 0.6729334 | 0.0644695 | 1.05932 | 2.200568 |
| cut4 | 1.879901 | 0.0996675 | 1.68457 | 2.075246 |
| /y2 cut1 | -2.16397 | 0.1480596 | 1.480596 | 1.877199 |
| cut2 | -0.9912152 | 0.0644245 | 1.05932 | 2.200568 |
| cut3 | -0.0230879 | 0.0687432 | 0.0230879 | 0.357154 |
| cut4 | 1.252164 | 0.0996675 | 1.68457 | 2.075246 |
| /y3 cut1 | -1.412372 | 0.0977772 | 1.604012 | -1.220733 |
| cut2 | -0.2300879 | 0.0687432 | 1.05932 | 2.200568 |
| cut3 | 0.8209522 | 0.0771653 | 0.8209522 | 0.9721935 |
| cut4 | 1.960042 | 0.1196586 | 1.731515 | 2.200568 |
| /y4 cut1 | -1.47999 | 0.0650596 | 1.607505 | -1.352476 |
| cut2 | -0.4218768 | 0.0443504 | 0.4218768 | 0.3349516 |
| cut3 | 0.172995 | 0.0432394 | 0.0882473 | 0.2577427 |
| cut4 | 0.9454906 | 0.0507422 | 0.8460376 | 1.044944 |
| var(e.SciAtt) | 0.5283629 | 0.0978703 | 0.3675036 | 0.7596315 |
Notes:

1. Our latent variable measures a negative attitude toward science, just as it did in [SEM] Example 35g.
2. In this MIMIC model, we allow males and females to have different underlying attitudes toward science.
3. The coefficient for SciAtt <- sex is −0.234. Variable sex is equal to 1 for females, thus females have a lower mean value for SciAtt by 0.234. Because our SciAtt measure is reversed, this means that females have a more positive attitude toward science. The effect is significant at better than the 1% level.
4. The difference between males and females in SciAtt is −0.234. Is that big or small, practically speaking?

In the ordered probit specification, predicted agreement with questions is determined by an index into a \(N(0, 1)\) distribution. The value of the index is then compared with the cutpoints to determine the probability that the response is 1, 2, 3, 4, or 5.

For statement 1, the loading on SciAtt is 1, and therefore the average difference in the probit index for males and females is \(-0.234 \times 1 = -0.234\) standard deviation units. Females are 0.234 standard deviations to the left of males on average.

For statement 2, the loading on SciAtt is 1.41, and therefore the average difference in the probit index is \(-0.234 \times 1.41 = -0.33\) standard deviation units.

You can work out the effect size for the other statements. We would say that the effect is medium sized.

**Fitting the model with the Builder**

Use the diagram in *Fitting the MIMIC model* above for reference.

1. Open the dataset.
   
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/gsem_issp93
   ```

2. Open a new Builder diagram.
   
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation*.

3. Put the Builder in gsem mode by clicking on the \(\mathcal{G}\) button.

4. Create the measurement component for SciAtt.

   Select the Add measurement component tool, \(\mathcal{M}\), and then click in the diagram halfway down and slightly left of the center.

   In the resulting dialog box,
   a. change the *Latent variable name* to SciAtt;
   b. select y1, y2, y3, and y4 by using the *Measurement variables* control;
   c. check *Make measurements generalized*;
   d. select *Ordinal, Probit* in the *Family/Link* control;
   e. select *Down* in the *Measurement direction* control;
   f. click on OK.

   If you wish, move the component by clicking on any variable and dragging it.
5. Create the variable for the formative indicator of SciAtt.
   a. Select the Add observed variable tool, □, and then click in the diagram directly above the oval for SciAtt and about one-fourth of the way down from the top. After adding it, you can click inside the rectangle to move the variable if you wish.
   b. In the Contextual Toolbar, select sex with the Variable control.

6. Create the path from the formative indicator to SciAtt.
   a. Select the Add path tool, −.
   b. Click in the bottom of the sex rectangle (it will highlight when you hover over it), and drag a path to the top of the SciAtt oval (it will highlight when you can release to connect the path).

7. Clean up the direction of the error.
   The error on SciAtt is likely to have been created below the oval for SciAtt. Choose the Select tool, ✗, and then click in the SciAtt oval. Click on one of the Error rotation buttons, □□, in the Contextual Toolbar until the error is where you want it.

8. Clean up the location of the path.
   If you do not like where the path between sex and SciAtt has been connected to its variables, use the Select tool, ✗, to click on the path, and then simply click on where it connects to a rectangle or oval and drag the endpoint.

   Click on the Estimate button,  □□, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

   You can open a completed diagram in the Builder by typing . webgetsem gsem_mimic

Reference

Also see
[SEM] Example 10 — MIMIC model
[SEM] Example 35g — Ordered probit and ordered logit
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
Example 37g — Multinomial logistic regression

Description

With the data below, we demonstrate multinomial logistic regression, also known as multinomial logit, mlogit, and family multinomial, link logit:

```
. use https://www.stata-press.com/data/r16/gsem_sysdsn1
(Health insurance data)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_sysdsn1.dta
obs: 644 Health insurance data
vars: 12 28 Mar 2018 13:46
(_dta has notes)

variable name   type    format   label                  variable label
------------------------------------------------------------------------------
site            byte    %9.0g   study site (1-3)
patid           float   %9.0g   patient id
insure          byte    %9.0g   insure insurance type
age             float   %10.0g  NEMC (ISCNRD-IBIRTHD)/365.25
male            byte    %8.0g   NEMC PATIENT MALE
nonwhite        byte    %8.0g   race
noinsur0        byte    %8.0g   no insurance at baseline
noinsur1        byte    %8.0g   no insurance at year 1
noinsur2        byte    %8.0g   no insurance at year 2
ppd0            byte    %8.0g   prepaid at baseline
ppd1            byte    %8.0g   prepaid at year 1
ppd2            byte    %8.0g   prepaid at year 2
```

Sorted by: patid

_1. Data on health insurance available to 644 psychologically depressed subjects.
3. insure: 1=indemnity, 2=prepaid, 3=uninsured.

See _Structural models 6: Multinomial logistic regression_ in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Simple multinomial logistic regression model
- Multinomial logistic regression model with constraints
- Fitting the simple multinomial logistic model with the Builder
- Fitting the multinomial logistic model with constraints with the Builder
Simple multinomial logistic regression model

In a multinomial logistic regression model, there are multiple unordered outcomes. In our case, these outcomes are recorded in variable `insure`. This variable records three different outcomes—indemnity, prepaid, and uninsured—recorded as 1, 2, and 3. The model we wish to fit is

```
1b.insure
   multinomial
      logit
2.insure
   multinomial
      logit
3.insure
   multinomial
      logit
1.nonwhite
```

The response variables are 1.insure, 2.insure, and 3.insure, meaning `insure = 1` (code for indemnity), `insure = 2` (code for prepaid), and `insure = 3` (code for uninsured). We specified that `insure = 1` be treated as the mlogit base category by placing a `b` on 1.insure to produce 1b.insure in the variable box.

Notice that there are no paths into 1b.insure. We could just as well have diagrammed the model with a path arrow from the explanatory variable into 1b.insure. It would have made no difference.

In one sense, omitting the path is more mathematically appropriate, because multinomial logistic base levels are defined by having all coefficients constrained to be 0.

In another sense, drawing the path would be more appropriate because, even with `insure = 1` as the base level, we are not assuming that outcome `insure = 1` is unaffected by the explanatory variables. The probabilities of the three possible outcomes must sum to 1, and so any predictor that increases one probability of necessity causes the sum of the remaining probabilities to decrease. If a predictor $x$ has positive effects (coefficients) for both 2.insure and 3.insure, then increases in $x$ must cause the probability of 1.insure to fall.

The choice of base outcome specifies that the coefficients associated with the other outcomes are to be measured relative to that base. In multinomial logistic regression, the coefficients are logs of the probability of the category divided by the probability of the base category, a mouthful also known as the log of the relative-risk ratio.
We drew the diagram one way, but we could just as well have drawn it like this:

In fact, we could just as well have chosen to indicate the base category by omitting it entirely from our diagram, like this:

Going along with that, we could type three different commands, each exactly corresponding to one of the three diagrams:

```
. gsem (1b.insure) (2.insure 3.insure <- i.nonwhite), mlogit
. gsem (1b.insure 2.insure 3.insure <- i.nonwhite), mlogit
. gsem (2.insure 3.insure <- i.nonwhite), mlogit
```

In the command language, however, we would probably just type

```
. gsem (i.insure <- i.nonwhite), mlogit
```

See [SEM] Intro 3 for a complete description of factor-variable notation. It makes no difference which diagram we draw or which command we type.
This model can be fit using the command syntax by typing

```
gsem (i.insure <- i.nonwhite), mlogit
```

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-556.59502</td>
</tr>
<tr>
<td>1</td>
<td>-551.78935</td>
</tr>
<tr>
<td>2</td>
<td>-551.78348</td>
</tr>
<tr>
<td>3</td>
<td>-551.78348</td>
</tr>
</tbody>
</table>

Generalized structural equation model  Number of obs = 616
Response : insure
Base outcome : 1
Family : multinomial
Link : logit
Log likelihood = -551.78348

| Coef. Std. Err.  z   P>|z|   [95% Conf. Interval] |
|---------|------------------|-------|--------|---------------------|
| 1.insure (base outcome) |                   |       |        |                     |
| 2.insure 1.nonwhite | .6608212 .2157321 3.06 0.002 .2379942 1.083648 |
| _cons     | -.1879149 .0937644 -2.00 0.045 -.3716896 -.0041401 |
| 3.insure 1.nonwhite | .3779586 .407589 0.93 0.354 -.4209011 1.176818 |
| _cons     | -1.941934 .1782185 -10.90 0.000 -2.291236 -1.592632 |

Notes:
1. The above results say that nonwhites are more likely to have `insure = 2` relative to 1 than whites, and that nonwhites are more likely to have `insure = 3` relative to 1 than whites, which obviously implies that whites are more likely to have `insure = 1`.
2. For a three-outcome multinomial logistic regression model with the first outcome set to be the base level, the probability of each outcome is
   \[
   \Pr(y = 1) = \frac{1}{D} \\
   \Pr(y = 2) = \exp(X_2\beta_2)/D \\
   \Pr(y = 3) = \exp(X_3\beta_3)/D
   \]
   where \(D = 1 + \exp(X_2\beta_2) + \exp(X_3\beta_3)\).
3. For whites—that is, for `1.nonwhite = 0`—we have \(X_2\beta_2 = -0.1879\) and \(X_3\beta_3 = -1.9419\). Thus \(D = 1.9721\), and the probabilities for each outcome are 0.5071, 0.4202, and 0.0727. Those probabilities sum to 1. You can make the similar calculations for nonwhites—that is, for `1.nonwhite = 1`—for yourself.
Multinomial logistic regression model with constraints

Using the same data, we wish to fit the following model:

In the above, \texttt{insure} = 2 and \texttt{insure} = 3 have paths pointing to them from different sets of predictors. They share predictor \texttt{1.nonwhite}, but \texttt{insure} = 2 also has paths from \texttt{age} and \texttt{1.male}, whereas \texttt{insure} = 3 also has paths from the \texttt{site} variables. When we fit this model, we will not obtain estimates of the coefficients on \texttt{age} and \texttt{1.male} in the equation for \texttt{insure} = 3. This is equivalent to constraining the coefficients for \texttt{age} and \texttt{1.male} to 0 in this equation. In other words, we are placing a constraint that the relative risk of choosing \texttt{insure} = 3 rather than \texttt{insure} = 1 is the same for males and females and is the same for all ages.
This model can be fit using command syntax by typing

```
gsem (2.insure <- i.nonwhite age i.male) > (3.insure <- i.nonwhite i.site), mlogit
```

Iteration 0:  log likelihood = -555.85446
Iteration 1:  log likelihood = -541.20487
Iteration 2:  log likelihood = -540.85219
Iteration 3:  log likelihood = -540.85164
Iteration 4:  log likelihood = -540.85164

Generalized structural equation model  Number of obs  =  615
Response  :  insure
Base outcome  :  1
Family  :  multinomial
Link  :  logit
Log likelihood = -540.85164

|                | Coef. | Std. Err. |   z  |  P>|z|   | [95% Conf. Interval] |
|----------------|-------|-----------|------|-------|-----------------------|
| 1.insure       | (base outcome) |             |      |       |                       |
| 2.insure       |       |           |      |       |                       |
| 1.nonwhite     | .7219663 | .2184994  | 3.30 | 0.001 | .2937153 1.150217     |
| age            | -.0101291 | .0058972  | -1.72 | 0.086 | -.0216874 .0014292   |
| 1.male         | .5037961 | .1912717  | 2.63 | 0.008 | .1289104 .8786818    |
| _cons          | .1249932 | .2743262  | 0.46 | 0.649 | -.4126763 .6626627   |
| 3.insure       |       |           |      |       |                       |
| 1.nonwhite     | .0569646 | .4200407  | 0.14 | 0.892 | -.7663001 .8802293   |
| site 2         | -1.273576 | .4562854 | -2.79 | 0.005 | -2.167879 -.3792728  |
| site 3         | .0434253 | .3470773  | 0.13 | 0.900 | -.6368337 .7236843   |
| _cons          | -1.558258 | .2540157 | -6.13 | 0.000 | -2.056119 -1.060396  |

We could have gotten identical results from Stata’s mlogit command for both this example and the previous one. To fit the first example, we would have typed

```
mlogit insure i.nonwhite
```

To obtain the results for this second example, we would have been required to type a bit more:

```
constraint 1 [Uninsured]age = 0
constraint 2 [Uninsured]1.male = 0
constraint 3 [Prepaid]2.site = 0
constraint 4 [Prepaid]3.site = 0
.mlogit insure i.nonwhite age i.male i.site, constraints(1/4)
```

Having mlogit embedded in gsem, of course, also provides the advantage that we can combine the mlogit model with measurement models, multilevel models, and more. See [SEM] Example 41g for a two-level multinomial logistic regression with random effects.

**Fitting the simple multinomial logistic model with the Builder**

Use the first diagram in *Simple multinomial logistic regression model* above for reference.

1. Open the dataset.

   In the Command window, type

   ```
   use https://www.stata-press.com/data/r16/gsem_sysdsn1
   ```
2. Open a new builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.

3. Put the Builder in gsem mode by clicking on the \( \text{gsem} \) button.

4. Create the rectangles for each possible outcome of the multinomial endogenous variable.
   Select the Add observed variables set tool, \( \text{Add} \), and then click in the diagram about one-third of the way in from the right and one-fourth of the way up from the bottom.
   In the resulting dialog box,
   a. select the Select variables radio button (it may already be selected);
   b. check Make variables generalized responses;
   c. select Multinomial, Logit in the Family/Link control;
   d. select insure in the Variable control;
   e. select Vertical in the Orientation control;
   f. click on OK.
   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the independent variable.
   a. Select the Add observed variable tool, \( \text{Add} \), and then click in the diagram to the left of insure.
   b. In the Contextual Toolbar, type 1.nonwhite in the Variable control and press Enter.

6. Create the paths from the independent variable to the rectangles for outcomes insure = 2 and insure = 3.
   a. Select the Add path tool, \( \text{-} \).
   b. Click in the right side of the 1.nonwhite rectangle (it will highlight when you hover over it), and drag a path to the left side of the 2.insure rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the \( \text{-} \) tool, click in the right side of the 1.nonwhite rectangle and drag a path to the left side of the 3.insure rectangle.

7. Clean up the location of the paths.
   If you do not like where the paths have been connected to the rectangles, use the Select tool, \( \text{Select} \), to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.

8. Estimate.
   Click on the Estimate button, \( \text{Estimate} \), in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder by typing . webgetsem gsem_mlogit1
Fitting the multinomial logistic model with constraints with the Builder

Use the diagram in *Multinomial logistic regression model with constraints* above for reference.

1. Open the dataset.
   In the Command window, type
   
   . use https://www.stata-press.com/data/r16/gsem_sysdsn1
   
2. Open a new Builder diagram.
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation*.

3. Put the Builder in `gsem` mode by clicking on the \( \text{gm} \) button.

4. Create the rectangles for each possible outcome of the multinomial endogenous variable.
   Select the Add observed variables set tool, \( \text{Obv} \), and then click in the diagram about one-third of the way in from the right and one-fourth of the way up from the bottom.
   In the resulting dialog box,
   a. select the *Select variables* radio button (it may already be selected);
   b. check *Make variables generalized responses*;
   c. select Multinomial, Logit in the *Family/Link* control;
   d. select `insure` in the *Variable* control;
   e. select *Vertical* in the *Orientation* control;
   f. click on OK.
   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the independent variables.
   Select the Add observed variables set tool, \( \text{Obv} \), and then click in the diagram about one-third from the left and one-fourth from the bottom.
   In the resulting dialog box,
   a. select the *Select variables* radio button (it may already be selected);
   b. uncheck *Make variables generalized responses*;
   c. use the *Variables* control and select `age`;
   d. type `1.male 1.nonwhite` in the *Variables* control after `age` (typing `1.varname` rather than using the *** button to create them as `i.varname` factor variables prevents rectangles corresponding to the base categories for these binary variables from being created);
   e. include the levels of the factor variable `site` by clicking on the *** button next to the *Variables* control. In the resulting dialog box, select the *Factor variable* radio button, select *Main effect* in the *Specification* control, and select `site` in the *Variables* control for *Variable 1*. Click on Add to varlist, and then click on OK;
f. select **Vertical** in the *Orientation* control;
g. click on **OK**.

6. Create the paths from the independent variables to the rectangles for outcomes `insure = 2` and `insure = 3`.
   a. Select the Add path tool, `-`
   b. Click in the right side of the `age` rectangle (it will highlight when you hover over it), and drag a path to the left side of the `2.insure` rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the `-` tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the given outcome of the dependent variable:

   1. `male -> 2.insure`
   2. `nonwhite -> 2.insure`
   3. `nonwhite -> 3.insure`
   4. `site -> 3.insure`
   5. `site -> 3.insure`
   6. `site -> 3.insure`

7. Clean up the location of the paths.
   If you do not like where the paths have been connected to the rectangles, use the Select tool, ![Select tool](select.png), to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.

8. Estimate.
   Click on the **Estimate** button, ![Estimate button](estimate.png), in the Standard Toolbar, and then click on **OK** in the resulting *GSEM estimation options* dialog box.

   You can open a completed diagram in the Builder by typing
   ```bash
   . webgetsem sem_mlogit2
   ```

**Reference**


**Also see**

[SEM] **Example 35g** — Ordered probit and ordered logit
[SEM] **Example 41g** — Two-level multinomial logistic regression (multilevel)
[SEM] **Intro 5** — Tour of models
[SEM] **gsem** — Generalized structural equation model estimation command
Below we discuss random-intercept and random-slope models in the context of multilevel models, and specifically, 2-level models, although we could just as well use higher-level models (see \texttt{[SEM] Example 39g}). Some people refer to these models as random-effects models and as mixed-effects models.

To demonstrate random-intercept and random-slope models, we will use the following data:

\begin{verbatim}
. use https://www.stata-press.com/data/r16/gsem_nlsy
(NLSY 1968)
. describe
   Contains data from https://www.stata-press.com/data/r16/gsem_nlsy.dta
     obs:       2,763  NLSY 1968
     vars:      21
     29 Mar 2018 11:30
     (._dta has notes)

     variable name        storage  display value label variable label
                      type  format label
    idcode          int   %8.0g    NLS ID
    year           int   %8.0g  interview year
    birth_yr       byte  %8.0g    birth year
    age            byte  %8.0g  age in current year
    race           byte  %8.0g    racelbl race
    msp            byte  %8.0g  1 if married, spouse present
    nev_mar        byte  %8.0g  1 if never married
    grade          byte  %8.0g  current grade completed
    collgrad       byte  %8.0g  1 if college graduate
    not_smsa       byte  %8.0g  1 if not SMSA
    c_city         byte  %8.0g  1 if central city
    south          byte  %8.0g  1 if south
    ind_code       byte  %8.0g  industry of employment
    occ_code       byte  %8.0g    occupation
    union          byte  %8.0g  1 if union
    wks_ue         byte  %8.0g  weeks unemployed last year
    ttl_exp        float  %9.0g  total work experience
    tenure         float  %9.0g  job tenure, in years
    hours          int   %8.0g  usual hours worked
    wks_work       int   %8.0g  weeks worked last year
    ln_wage        float  %9.0g  ln(wage/GNP deflator)

Sorted by: idcode  year
\end{verbatim}

\texttt{. notes}
\texttt{. _dta:}
\begin{enumerate}
\item Data from National Longitudinal Survey of Young Women 14–27 years of age in 1968 (NLSY), Center for Human Resource Research, Ohio State University, first released in 1989.
\item These data were subsetted for purposes of demonstration.
\end{enumerate}
These 2-level data are recorded in long form, that is, each observation corresponds to a year within a subject and the full set of data is spread across repeated observations.

```
.list id year ln_wage union grade in 1/20, sepby(idcode)
```

<table>
<thead>
<tr>
<th>idcode</th>
<th>year</th>
<th>ln_wage</th>
<th>union</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1970</td>
<td>1.451214</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1971</td>
<td>1.02862</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1972</td>
<td>1.589977</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1973</td>
<td>1.780273</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1975</td>
<td>1.777012</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1977</td>
<td>1.778681</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1978</td>
<td>2.493976</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1980</td>
<td>2.551715</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1983</td>
<td>2.420261</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1985</td>
<td>2.614172</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1987</td>
<td>2.536374</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1988</td>
<td>2.462927</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1971</td>
<td>1.360348</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1972</td>
<td>1.206198</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1973</td>
<td>1.549883</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1975</td>
<td>1.832581</td>
<td>.</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1977</td>
<td>1.726721</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1978</td>
<td>1.689991</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1980</td>
<td>1.726964</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>1982</td>
<td>1.808289</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

In the repeated observations for a subject, some variables vary (they are at the observation level, such as `ln_wage`) and other variables do not vary (they are at the subject level, such as `grade`).

When using `gsem`, multilevel data must be recorded in the long form except in one case. The exception is latent growth curve models, which can be fit in the long or wide form. In the wide form, there is one physical observation for each subject and multiple variables within subject, such as `ln_wage_1970`, `ln_wage_1971`, and so on. Researchers from a structural equation modeling background think about latent growth models in the wide form; see [SEM] Example 18.

In all other cases, if your data are in the wide form, use Stata’s `reshape` command (see [D] `reshape`) to convert the data to long form.

See Structural models 1: Linear regression and Multilevel mixed-effects models in [SEM] Intro 5 for background.

## Remarks and examples

Remarks are presented under the following headings:

- Random-intercept model, single-equation formulation
- Random-intercept model, within-and-between formulation
- Random-slope model, single-equation formulation
- Random-slope model, within-and-between formulation
- Fitting the random-intercept model with the Builder
- Fitting the random-slope model with the Builder
Random-intercept model, single-equation formulation

There are two formulations of the random-intercept model, which we call the single-equation formulation and the within-and-between formulation. Results from both formulations are identical unless you have observations with missing values, in which case the within-and-between formulation will sometimes use more of the data.

We will show you both formulations, but the single-equation formulation makes a good starting point. The model we wish to fit is

\[
\text{ln}_\text{wage} \varepsilon_1 \text{grade}\text{1.union}\text{idcode1}
\]

We use factor-variable notation in the diagram above; see [SEM] Example 37g.

We are using multilevel data. \text{ln}_\text{wage} and \text{union} (union membership) vary at the observation level, while \text{grade} (school completion) varies at the subject level. We have used shading to emphasize that.

In this model, we are including a random intercept (a random effect) at the subject level. Double-ringed \text{idcode} is saying, “I am a latent variable at the \text{idcode} level—meaning I am constant within identification codes and vary across identification codes—and I correspond to a latent variable named \text{M1}.” The \text{M1} part of the statement came from the subscript 1; the \text{M} part is fixed.

Double-ringed \text{idcode} indicates a latent variable constant within \text{idcode}—a random effect. And the fact that the path from the latent variable is pointing to a box and not to another path means that the latent variable is used as a random intercept rather than a random slope. By the way, variable \text{idcode} in the data contains each subject’s identification number.

Using command syntax, we can fit this model by typing

\[
.\text{gsem (ln}_\text{wage} \leftarrow \text{i.union grade M1[idcode])}
\]

Fitting fixed-effects model:
Iteration 0:  log likelihood = -925.06629
Iteration 1:  log likelihood = -925.06629
Refining starting values:
Grid node 0:  log likelihood = -763.3769
Fitting full model:
Iteration 0:  log likelihood = -763.3769
Iteration 1:  log likelihood = -622.04625 (backed up)
Iteration 2:  log likelihood = -613.54948
Iteration 3:  log likelihood = -607.56242
Iteration 4:  log likelihood = -607.49246
Iteration 5:  log likelihood = -607.49233
Iteration 6:  log likelihood = -607.49233
Example 38g — Random-intercept and random-slope models (multilevel) 399

Generalized structural equation model  Number of obs    =    1,904
Response    :  ln_wage
Family       :  Gaussian
Link         :  identity
Log likelihood =  -607.49233
( 1)  [ln_wage]M1[idcode] = 1

|                  | Coef.  | Std. Err. |    z  |   P>|z|  | [95% Conf. Interval] |
|------------------|--------|-----------|------|------|---------------------|
| ln_wage          |        |           |      |      |                     |
| 1.union          | .1637408 | .0227254 |  7.21|  0.000| .1191998 -- .2082818|
| grade            | .0767919 | .0067923 | 11.31|  0.000| .0634791 -- .0901046|
| M1[idcode]       | 1 (constrained) |     |      |      |                     |
| _cons            | .7774129 | .0906282 |  8.58|  0.000| .5997848 -- .955041  |
| var(M1[idcode])  | .080247 | .0073188 |  0.000|      | .0671113 -- .0959537|
| var(e.ln_w~e)    | .078449 | .0028627 |  0.000|      | .0730342 -- .0842653|

Notes:

1. The ln_wage <- M1[idcode] coefficient is constrained to be 1. Such constraints are automatically supplied by gsem to identify the latent variable. Our model is

   \[ \text{ln}_wage = \cdots + \beta_3 \times \text{M1[idcode]} + \text{cons} \]

Thus M1[idcode] is being used as a random intercept.

   Remember that the square bracketed [idcode] means that M1 is constant within idcode and varies only across idcode.

2. The variance of our random intercept is estimated to be 0.0802, which is greater than the estimated error variance of 0.0784.

3. Although it is obvious in this case that the latent variable (random intercept) has sufficient variance that it cannot be ignored, we can test whether the variance is large enough that we could not ignore it. The test will be up against a boundary (variances cannot be less than 0), and so the test will be conservative. To perform the test, we would type

   \[
   \begin{align*}
   &. gsem (\text{ln}_wage <- i.\text{union grade}) & \text{// fit model w/o random intercept} \\
   &. \text{estimates store without} \\
   &. gsem (\text{ln}_wage <- i.\text{union grade M1[idcode]}) & \text{// fit model w/ random intercept} \\
   &. \text{estimates store with} \\
   &. \text{lrttest without with} & \text{// order of arguments unimportant}
   \end{align*}
   \]
Random-intercept model, within-and-between formulation

The other way to write the random-intercept model is

\[
\begin{align*}
\text{ln}_{\text{wage}} &= \epsilon_1 + \text{grade} + \epsilon_2, \\
1.\text{union} &= \text{idcode}, \\
\text{ln}_{\text{wage}} &= \epsilon_1.
\end{align*}
\]

Figure 2

Do not read \textit{grade} pointing to double-ringed \textit{idcode} as \textit{grade} being a predictor of \textit{idcode}. That would make no sense. Double rings indicate a latent variable, and \textit{grade} is a predictor of a latent variable. In particular, the subscript 1 on \textit{idcode} indicates that the latent variable is named M1. Thus \textit{grade} is a predictor of M1. The \textit{idcode} inside the double rings says that M1 is constant within \textit{idcode}. Thus \textit{grade}, which itself does not vary within \textit{idcode}, is a predictor of M1, which does not vary within \textit{idcode}; said more elegantly, \textit{grade} is a predictor of M1 at the subject level.

It is logically required that \textit{grade} vary at the same or higher level as M1[idcode], and \texttt{gsem} will check that requirement for you.

In this model, M1[idcode] contains both the random intercept and the grade effect. There is now an equation for M1[idcode], with an error associated with it, and it will be the variance of the error term that will reflect the variance of the random intercept.
To fit this within-and-between formulation of our model, we type

```
gsem (ln_wage <- i.union M1[idcode]) (M1[idcode] <- grade)
```

Fitting fixed-effects model:
Iteration 0: log likelihood = -1091.655
Iteration 1: log likelihood = -1091.655
Refining starting values:
Grid node 0: log likelihood = -886.50236
Fitting full model:
Iteration 0: log likelihood = -886.50236 (not concave)
Iteration 1: log likelihood = -683.7337
Iteration 2: log likelihood = -630.91327
Iteration 3: log likelihood = -608.14536
Iteration 4: log likelihood = -607.49376
Iteration 5: log likelihood = -607.49233
Iteration 6: log likelihood = -607.49233

Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>ln_wage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
<tr>
<td>Log likelihood</td>
<td>-607.49233</td>
</tr>
</tbody>
</table>

(1) [ln_wage]M1[idcode] = 1

|   | Coef.  | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|---|--------|-----------|-------|------|---------------------|
| ln_wage |        |           |       |      |                     |
| 1.union | .1637408 | .0227254  | 7.21  | 0.000 | .1191998 .2082818  |
| M1[idcode] | 1 (constrained) |
| _cons   | .7774129 | .0906282  | 8.58  | 0.000 | .5997848 .955041  |
| M1[idcode] | grade |        |       |      |                     |
|        | .0767919 | .0067923  | 11.31 | 0.000 | .0634791 .0901046 |
| var(e.M1[idcode]) |        |           |       |      |                     |
|        | .080247  | .0073188  |       |      | .0671113 .0959537 |
| var(e.ln_w-e) |        |           |       |      |                     |
|        | .078449  | .0028627  |       |      | .0730342 .0842653 |

Notes:
1. Results are identical to what we previously obtained.
2. The within-and-between formulation is equivalent to the single-equation formulation if there are no missing values in the data.
3. In this simple model, the two formulations are also equivalent even in the presence of missing values.
4. If M1[idcode] were also being used to predict another endogenous variable, then missing values in grade would only cause the equation for the other endogenous variable to have to omit those observations in the within-and-between formulation.
Random-slope model, single-equation formulation

Let us now turn to random slopes. Because it is generally—not always—a good idea to include random intercepts with random slopes, we are going to include both. In addition, we are going to complicate our model by adding an interaction term between union membership and grade so we can make another point that we will not explain until the next section.

The model we wish to fit is

\[
\ln_{\text{wage}} = \beta_3 + \beta_4 \times M_2[id\text{code}] \times 1.\text{union} + \epsilon_1
\]

Figure 3. Demonstrated formulation 1

Ignore the interaction between union membership and grade. We could have omitted it from this example, but we want to show later the strikingly different way that interaction can be written in the within-and-between formulation.

In this model, we have double-ringed idcode, just as we did in a previous section introducing a random intercept into the model. This time, however, we also have a second double-ringed idcode adding a random slope. When a path from a latent variable points to another path, it is specifying a random slope. When specifying a random slope for a variable, you still include the variable in the usual, fixed-slope way, and then you add the extra component.

The result of the extra component will be to add an interaction into the model. 1.union will affect ln_wage, just as we have drawn, and the extra component will add 1.union \times M_2[id\text{code}]. That is, the effect of 1.union in our model will be

\[
\ln_{\text{wage}} = \beta_3 + \beta_4 \times M_2[id\text{code}] \times 1.\text{union} + \epsilon_1
\]

The fixed-plus-random, total slope is \( \beta_3 + \beta_4 \times M_2[id\text{code}] \). Latent variable M2 has an arbitrary scale, and therefore coefficient \( \beta_4 \) cannot be identified. So constraining \( \beta_4 = 1 \) results in the total slope being \( \beta_3 + M_2[id\text{code}] \), which is exactly how you would expect a fixed-plus-random slope to look. In other words, drawing the path from idcode_2 to the path with coefficient \( \beta_3 \) indicates that M2[id\text{code}] is added to \( \beta_3 \).
To fit this model in the command language, we type

```stata
.gsem (ln_wage <- i.union grade i.union#c.grade M1[idcode] 1.union#M2[idcode])
```

Fitting fixed-effects model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-925.06629</td>
</tr>
<tr>
<td>1</td>
<td>-925.06629</td>
</tr>
</tbody>
</table>

Refining starting values:

<table>
<thead>
<tr>
<th>Grid node</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-869.92256</td>
</tr>
</tbody>
</table>

Fitting full model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-869.92256 (not concave)</td>
</tr>
<tr>
<td>1</td>
<td>-727.15806 (not concave)</td>
</tr>
<tr>
<td>2</td>
<td>-711.74718 (not concave)</td>
</tr>
<tr>
<td>3</td>
<td>-684.33867 (not concave)</td>
</tr>
<tr>
<td>4</td>
<td>-665.94123 (not concave)</td>
</tr>
<tr>
<td>5</td>
<td>-610.14526</td>
</tr>
<tr>
<td>6</td>
<td>-589.89989</td>
</tr>
<tr>
<td>7</td>
<td>-582.24119</td>
</tr>
<tr>
<td>8</td>
<td>-581.298</td>
</tr>
<tr>
<td>9</td>
<td>-581.29004</td>
</tr>
<tr>
<td>10</td>
<td>-581.29003</td>
</tr>
</tbody>
</table>

Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>Family</th>
<th>Link</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>ln_wage</td>
<td>Gaussian</td>
<td>identity</td>
<td>-581.29003</td>
</tr>
</tbody>
</table>

(1) [ln_wage]M1[idcode] = 1
(2) [ln_wage]1.union#M2[idcode] = 1

| Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|-----------------|-----------------|-----------------|-----------------|
| ln_wage         |                 |                 |                 |
| 1.union grade   | .1199049 .1508189 0.80 0.427 -.1756946 .4155045 |
| grade           | .0757883 .0081803 9.26 0.000 .0597552 .0918215 |
| union# c.grade  | .0019983 .0113534 0.18 0.860 -.020254 .0242506 |
| M1[idcode]      | 1 (constrained) |
| union# M2[idcode] | 1 (constrained) |
| _cons           | .7873884 .1086476 7.25 0.000 .574443 1.000334 |

| Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|-----------------|-----------------|-----------------|-----------------|
| var(M1[idcode]) | .0927931 .0088245 0.0770136 .1118056 |
| var(M2[idcode]) | .0823065 .018622 .052826 .1282392 |

| Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|-----------------|-----------------|-----------------|-----------------|
| cov(M1[idcode], M2[idcode]) | -.0549821 .0116103 -4.74 0.000 -.077738 -.032263 |

| Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|-----------------|-----------------|-----------------|-----------------|
| var(e.ln_w~e)  | .0720873 .0027135 .0669603 .0776068 |
Notes:

1. $M_1[idcode]$ is the random intercept. The coefficient on it is constrained to be 1, just as previously and just as we would expect.

2. The coefficient on $1.union$ is the fixed part of the slope of $union$.

3. $M_2[idcode]$ is the random part of the slope of $union$.

   The coefficient on $1.union#M_2[idcode]$ is constrained to be 1, just as we would expect. $1.union#M_2[idcode]$ is the way Stata writes $1.union \times M_2[idcode]$.

4. There is an unexpected term in the output, $0.union#M_2[idcode]$, shown with coefficient 0. The first thing to remember about unexpected terms is that they are irrelevant if their coefficients are 0. $gsem$ reports the coefficient as being 0 (omitted), which is $gsem$'s way of saying, “Here is a line that I did not even include in the model.” There are a lot of terms $gsem$ could tell us about that were not included in the model, so why did $gsem$ feel obligated to tell us about this term? The term has to do with how Stata tracks base levels of factor variables.

   There is a setting—set showbaselevels off—that will prevent lines like that from being displayed. There is also a setting—set showbaselevels all—that will show even more of them! The default is set showbaselevels on.

5. We specified the interaction as $1.union#M_2[idcode]$ rather than $i.union#M_2[idcode]$. Even so, using $#.union#M_2[idcode]$ or $i.union#M_2[idcode]$ makes no difference because $union$ takes on two values. If $union$ took on three values, however, think about how we would diagram the model. We would have two latent variables, and we would want $1.union#M_2[idcode]$ and $2.union#M_3[idcode]$. If $union$ took on three or more values, typing $i.union#M_2[idcode]$ simply would not produce the desired result.

**Random-slope model, within-and-between formulation**

To properly demonstrate the within-and-between formulation, we need a more complicated model; we made one in the previous section when we included $1.union#c.grade$ and then told you to ignore it.

Pretend that we had omitted $1.union#c.grade$ from the random-slope, single-equation formulation model. It would have looked like this:

![Figure 4. Simple formulation 1](image-url)
Let’s call this model Simple formulation 1 as compared with the model that we fit in the previous section, which we will call Demonstrated formulation 1. We could have fit Simple formulation 1 by typing

```
gsem (ln_wage <- i.union grade M1[idcode] 1.union#M2[idcode])
```

The corresponding within-and-between model, which we will call Simple formulation 2, would look something like this:

![Diagram of Simple formulation 2](image)

Figure 5. Simple formulation 2

We say that it would look something like the above because Simple formulations 1 and 2 are not identical models. In particular, there is nothing in Simple formulation 2 that corresponds to the correlation between $M1[idcode]$ and $M2[idcode]$ in Simple formulation 1. Other than that, the models are identical. In Simple formulation 1, *grade* directly affects $\ln(wage)$. In Simple formulation 2, *grade* affects $M1[idcode]$ and $M1[idcode]$ affects $\ln(wage)$. The results are the same either way. Many researchers prefer Simple formulation 2 because it visually separates the within (subject) and between (subject) levels.

But as we said, the two formulations are not identical. They would be identical if we added a curved covariance path between $\epsilon_2$ and $idcode_2$, but Stata will not allow us to do that.

On the other hand, if you wanted to fit the simple model without the correlation, you could do it in either formulation. For Simple formulation 1, you would type

```
gsem (ln_wage <- i.union grade M1[idcode] 1.union#M2[idcode]), ///
       cov(M1[idcode]*M2[idcode]@0)
```

For Simple formulation 2, you would type

```
gsem (ln_wage <- i.union M1[idcode] 1.union#M2[idcode]) (M1[idcode] <- grade)
```

Either way, results would be identical.
In more complicated models, you can correlate the random intercepts and slopes even in a second formulation. Consider Demonstrated formulation 2:

\[
\begin{align*}
\text{ln}_w &= \epsilon_1 + 1.\text{union} + \text{idcode}_1 \\
\text{idcode}_2 &= \epsilon_3 \\
\text{idcode}_1 &= \epsilon_2 \\
\text{grade} &\rightarrow 1.\text{union} \\
\text{1.union} &\rightarrow \ln_w \\
\end{align*}
\]

Figure 6. Demonstrated formulation 2

In this model, we allow correlation between \(\epsilon_3\) and \(\epsilon_2\)—which Stata does allow—and that is the same as allowing correlation between \(M1[idcode]\) and \(M2[idcode]\) in the formulation-1 model. The general rule is that formulation 2 can be used to correlate the random intercepts and random slopes when both \(M1\) and \(M2\) are (latent) exogenous or (latent) endogenous.

Demonstrated formulation 2 corresponds to Demonstrated formulation 1. In Demonstrated formulation 1, we included \text{l}_n\text{w}_a_g_e <- i.\text{union}#c.\text{grade}. Do you see how that term is implied by the path diagram above? Variable grade affects \(M2[idcode]\), which in turn affects the slope of 1.union. Affecting the slope is the same as an interaction.

And now you know why we complicated Demonstrated formulation 1. We did that so that we can show you Demonstrated formulation 2:

```stata
. gsem (ln_wage <- i.union M1[idcode] 1.union#M2[idcode])
   > (M1[idcode] M2[idcode] <- grade), cov(e.M1[idcode]*e.M2[idcode])
```

Fitting fixed-effects model:
Iteration 0: log likelihood = -1091.655
Iteration 1: log likelihood = -1091.655

Refining starting values:
Grid node 0: log likelihood = -982.18261

Fitting full model:
Iteration 0: log likelihood = -982.18261 (not concave)
Iteration 1: log likelihood = -690.60714 (not concave)
Iteration 2: log likelihood = -677.13819 (not concave)
Iteration 3: log likelihood = -661.26658
Iteration 4: log likelihood = -615.66049
Iteration 5: log likelihood = -585.69402
Iteration 6: log likelihood = -581.34158
Iteration 7: log likelihood = -581.29005
Iteration 8: log likelihood = -581.29003
Example 38g — Random-intercept and random-slope models (multilevel) 407

Generalized structural equation model
Response : ln_wage
Family : Gaussian
Link : identity
Log likelihood = -581.29003

( 1) [ln_wage]M1[idcode] = 1
( 2) [ln_wage]1.union#M2[idcode] = 1

|                        | Coef.  | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|------------------------|--------|-----------|-------|-------|----------------------|
| ln_wage                |        |           |       |       |                      |
| 1.union                | .119905| .1508188  | 0.80  | 0.427 | -.1756945 .4155045   |
| M1[idcode]             | 1      |           |       |       | (constrained)        |
| union#M2[idcode]       |        |           |       |       |                      |
| 1                     | 1      |           |       |       | (constrained)        |
| _cons                  | .7873884| .1086476 | 7.25  | 0.000 | .5744431 1.000334    |
| M1[idcode] grade       | .0757883| .0081803 | 9.26  | 0.000 | .0597552 .0918215    |
| M2[idcode] grade       | .0019983| .0113534 | 0.18  | 0.860 | -.020254 .0242506    |
| var(e.M1[idcode])      | .092793| .0088245  |       |       | .0770136 .1118055    |
| var(e.M2[idcode])      | .0823065| .018622  |       |       | .052826 .1282392     |
| cov(e.M1[idcode],e.M2[idcode]) | -.0549822| .0116103 | -4.74 | 0.000 | -.077738 -.032263    |
| var(e.ln_w-e)          | .0720873| .0027135 |       |       | .0669603 .0776068    |

Note:
1. Results of the above are identical to the results of Demonstrated formulation 1.

Fitting the random-intercept model with the Builder

Use the diagram in Random-intercept model, single-equation formulation above for reference.
1. Open the dataset.
   In the Command window, type
   . use https://www.stata-press.com/data/r16/gsem_nlsy

2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.
3. Put the Builder in gsem mode by clicking on the $\text{gsem}$ button.
4. Create the endogenous variable.
   a. Select the Add observed variable tool, □, and then click in the diagram about one-third of
      the way in from the right and one-third of the way up from the bottom. After adding it, you
      can click inside the rectangle and move the variable if you wish.
   b. In the Contextual Toolbar, select ln_wage with the Variable control.

5. Create the observed exogenous variables.
   Select the Add observed variables set tool, □□□, and then click in the diagram about one-third of
   the way in from the left and one-third of the way up from the bottom.
   In the resulting dialog box,
   a. select the Select variables radio button (it may already be selected);
   b. use the Variables control and select grade;
   c. type 1.union in the Variables control after grade (typing 1.union rather than using the
      button to create i.union prevents the rectangle corresponding to the base category for
      this binary variable from being created);
   d. select Vertical in the Orientation control;
   e. click on OK.
   If you wish, move the set of variables by clicking on any variable and dragging it.

6. Create the multilevel latent variable corresponding to the random intercept.
   a. Select the Add multilevel latent variable tool, ⊙, and click above the rectangle for grade.
   b. In the Contextual toolbar, click on the □ button.
   c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control
      and selecting idcode > Observations in the next control.
   d. Specify M1 as the Base name.
   e. Click on OK.

7. Create the paths from the exogenous variables to ln_wage.
   a. Select the Add path tool, .
   b. Click in the right side of the 1.union rectangle (it will highlight when you hover over it),
      and drag a path to the left side of the ln_wage rectangle (it will highlight when you can
      release to connect the path).
   c. Continuing with the . tool, draw paths from the right side of the grade rectangle to the
      left side of the ln_wage rectangle and from the right side of the idcode1 double oval to
      the left side of the ln_wage rectangle.

8. Clean up the location of the paths.
   If you do not like where the paths have been connected to the rectangles or oval, use the Select
   tool, , to click on the path, and then simply click on where it connects to a rectangle or oval
   and drag the endpoint.

   Click on the **Estimate** button, \( \mathbb{E} \), in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

   You can open a completed diagram in the Builder by typing
   
   \[ .\hspace{1em} \text{webgetsem gsem_rint} \]

**Fitting the random-slope model with the Builder**

   Use the diagram in *Random-slope model, single-equation formulation* above for reference.

1. Open the dataset.

   In the Command window, type
   
   \[ .\hspace{1em} \text{use https://www.stata-press.com/data/r16/gsem_nlsy} \]

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in **gsem** mode by clicking on the \( \mathbb{G} \) button.

4. Increase the width of the observed variable rectangles to accommodate the length of the name of the interaction term.

   From the SEM Builder menu, select **Settings > Variables > All observed....**

   In the resulting dialog box, change the first size to 1 and click on **OK**.

5. Create the endogenous variable.

   a. Select the Add observed variable tool, \( \mathbb{O} \), and then click in the diagram about one-third of the way in from the right and one-third of the way up from the bottom. After adding it, you can click inside the rectangle and move the variable if you wish.

   b. In the Contextual Toolbar, select \( \ln\text{wage} \) with the **Variable** control.

6. Create the observed exogenous variables.

   Select the Add observed variables Set tool, \( \mathbb{S} \), and then click in the diagram about one-third of the way in from the left and one-third of the way up from the bottom.

   In the resulting dialog box, change the first size to 1 and click on **OK**.

   a. Select the **Select variables** radio button (it may already be selected);

   b. type \( \text{1.union} \) in the **Variables** control (typing \( \text{1.union} \) rather than using the \( \mathbb{S} \) button to create \( \text{i.union} \) prevents the rectangle corresponding to the base category for this binary variable from being created);

   c. use the **Variables** control and select \( \text{grade} \);

   d. type \( \text{1.union#c.grade} \) in the **Variables** control after \( \text{grade} \);

   e. select **Vertical** in the **Orientation** control;

   f. click on **OK**.

   If you wish, move the set of variables by clicking on any variable and dragging it.
6. Create the multilevel latent variable corresponding to the random intercept.
   a. Select the Add multilevel latent variable tool, \( \text{Add multilevel latent variable} \), and click above the rectangle for \( \ln \text{wage} \).
   b. In the Contextual Toolbar, click on the button.
   c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting \text{idcode} > Observations in the next control.
   d. Specify \text{M1} as the Base name.
   e. Click on OK.

7. Create the paths from the exogenous variables to \( \ln \text{wage} \).
   a. Select the Add path tool, \( \text{Add path} \).
   b. Click in the right side of the 1.union rectangle (it will highlight when you hover over it), and drag a path to the left side of the \( \ln \text{wage} \) rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the \( \text{Add path} \) tool, draw paths from the right sides of the grade and 1.union#c.grade rectangles to the left side of the \( \ln \text{wage} \) rectangle and from the bottom of the idcode\text{1} double oval to the top of the \( \ln \text{wage} \) rectangle.

8. Create the random slope.
   a. Select the Add multilevel latent variable tool, \( \text{Add multilevel latent variable} \), and click above the path from 1.union to \( \ln \text{wage} \).
   b. In the Contextual Toolbar, click on the button.
   c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting \text{idcode} > Observations in the next control.
   d. Specify \text{M2} as the Base name.
   e. Click on OK.
   f. Select the Add path tool, \( \text{Add path} \).
   g. Click in the bottom of the idcode\text{2} double oval, and drag a path to the path between 1.union and \( \ln \text{wage} \).

9. Create the covariance between the random slope and random intercept.
   a. Select the Add covariance tool, \( \text{Add covariance} \).
   b. Click in the top-right quadrant of the idcode\text{2} double oval, and drag a covariance to the top left of the idcode\text{1} double oval.

10. Clean up paths and covariance.
    If you do not like where a path has been connected to its variables, use the Select tool, \( \text{Select} \), to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint. Similarly, you can change where the covariance connects to the latent variables by clicking on the covariance and dragging the endpoint. You can also change the bow of the covariance by clicking on the covariance and dragging the control point that extends from one end of the selected covariance.
11. Estimate.

Click on the Estimate button, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder by typing

.webgetsem gsem_rslope

Reference


Also see

[SEM] Example 39g — Three-level model (multilevel, generalized response)
[SEM] Example 42g — One- and two-level mediation models (multilevel)
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
Example 39g — Three-level model (multilevel, generalized response)

Description

To demonstrate three-level models, we use the following data:

```
. use https://www.stata-press.com/data/r16/gsem_melanoma
(Skin cancer (melanoma) data)
. describe
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>label</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>nation</td>
<td>byte</td>
<td>%12.0g</td>
<td>nation</td>
<td>Nation ID</td>
<td></td>
</tr>
<tr>
<td>region</td>
<td>byte</td>
<td>%9.0g</td>
<td>Region ID</td>
<td>EEC level-I areas</td>
<td></td>
</tr>
<tr>
<td>county</td>
<td>int</td>
<td>%9.0g</td>
<td>County ID</td>
<td>EEC level-II/level-III areas</td>
<td></td>
</tr>
<tr>
<td>deaths</td>
<td>int</td>
<td>%9.0g</td>
<td>No. deaths during 1971-1980</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expected</td>
<td>float</td>
<td>%9.0g</td>
<td>No. expected deaths</td>
<td></td>
<td></td>
</tr>
<tr>
<td>uv</td>
<td>float</td>
<td>%9.0g</td>
<td>UV dose, mean-centered</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sorted by: nation region county

```
. notes
```

_dta:
2. Data on 7 nations, 3-95 regions w/i nation, 1-13 counties w/i region.
4. Variable expected contains # of expected male deaths based on crude rates for the combined counties.

Rabe-Hesketh and Skrondal (2012, exercise 13.7) describe data from the _Atlas of Cancer Mortality in the European Economic Community_ (EEC) (Smans, Mair, and Boyle 1993). The data were analyzed in Langford, Bentham, and McDonald (1998) and record the number of deaths among males due to malignant melanoma during 1971–1980.

Data are stored in the long form. Observations are counties within regions within nation. These data and some of the models fit below are also demonstrated in [ME] _membreg_.

See _Structural models 4: Count models_ and _Multilevel mixed-effects models_ in [SEM] _Intro 5_ for background.
Remarks and examples

Remarks are presented under the following headings:

Three-level negative binomial model
Three-level Poisson model
Testing for overdispersion
Fitting the models with the Builder

Three-level negative binomial model

The model we wish to fit is

Deaths due to malignant melanoma at the county level are modeled as being affected by ultraviolet exposure with random region and nation effects.
To fit this model, we type

```
  . gsem (deaths <- uv M1[nation] M2[nation>region]), nbreg exposure(expected)
```

Fitting fixed-effects model:

| Iteration | log likelihood | z  | P>|z|  | [95% Conf. Interval] |
|-----------|----------------|----|-------|-----------------------|
| 0         | -1361.855      |    |       |                       |
| 1         | -1230.0211     |    |       |                       |
| 2         | -1211.049      |    |       |                       |
| 3         | -1202.5641     |    |       |                       |
| 4         | -1202.5329     |    |       |                       |
| 5         | -1202.5329     |    |       |                       |

Refining starting values:

Grid node 0: log likelihood = -1209.6951

Fitting full model:

| Iteration | log likelihood | z  | P>|z|  | [95% Conf. Interval] |
|-----------|----------------|----|-------|-----------------------|
| 0         | -1209.6951     |    |       | (not concave)         |
| 1         | -1195.0761     |    |       | (not concave)         |
| 2         | -1189.7235     |    |       | (not concave)         |
| 3         | -1167.58       |    |       | (not concave)         |
| 4         | -1145.4325     |    |       | (not concave)         |
| 5         | -1138.4471     |    |       |                       |
| 6         | -1088.3882     |    |       |                       |
| 7         | -1086.7992     |    |       |                       |
| 8         | -1086.4085     |    |       |                       |
| 9         | -1086.3903     |    |       |                       |
| 10        | -1086.3902     |    |       |                       |
| 11        | -1086.3902     |    |       |                       |

Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>Number of obs</th>
</tr>
</thead>
<tbody>
<tr>
<td>deaths</td>
<td>354</td>
</tr>
</tbody>
</table>

Family      : nbinomial
Dispersion  : mean
Link        : log

Log likelihood = -1086.3902

( 1) [deaths]M1[nation] = 1
( 2) [deaths]M2[nation>region] = 1

|                  | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------------|-------|-----------|-------|-------|-----------------------|
| deaths uv        | -.0335933 | .0113725  | -2.95 | 0.003 | -.055883 -.0113035    |
| M1[nation]       | 1 (constrained) |          |       |       |                       |
| M2[nation>region]| 1 (constrained) |          |       |       |                       |
| _cons            | -.0790606 | .1295931  | -0.61 | 0.542 | -.3330583 .1749372    |
| ln(expected)     | 1 (exposure) |          |       |       |                       |
|                  |        |           |       |       |                       |
| /deaths lnp alpha| -.4182603 | .3415036  | -4.851937 | -3.513268    |

|                  | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------------|-------|-----------|-------|-------|-----------------------|
| var( M1[nation]) | .1283614 | .0678971  |   .0455187 | .3619758    |
| var( M2[nation>region]) | .0401818 | .0104855  | .0240938 | .067012    |
Notes:

1. This is a three-level model of counties nested within region nested within nation, so we specified the latent variables as $M_1[nation] M_2[nation>region]$. Actually, we did the same thing in the diagram when we used the SEM Builder to define the latent variables, but the nesting information does not show in the double rings.

2. We fit this model by using negative binomial regression, also known as a mean-dispersion model. In the command, we typed `nbreg`, which is shorthand for `family(nbinomial mean) link(log)`.

3. A negative binomial distribution can be regarded as a gamma mixture of Poisson random variables, where said gamma distribution has mean 1 and variance $\alpha$. The estimated $\ln(\alpha)$ is $-4.183$, which is small; $\alpha$ is estimated as 0.0153.

4. Zero does not mean lack of overdispersion, because we are including random effects that also allow for extra dispersion. For a discussion on these issues, see [ME] `menbreg`.

5. Notice that we specified `exposure(expected)`, where variable `expected` contains the expected number of deaths based on crude rates.

   The `exposure()` option is allowed with Poisson and negative binomial models. If we specify `exposure(varname)`, we are usually saying that each observation’s time at risk is recorded in variable `varname`. When we omit the option, we are saying that each observation has the same time at risk. Obviously, if one observation had twice the time at risk of another observation, but was otherwise identical, we would expect twice the number of events in the first observation.

   In this case, however, we are using `exposure()` differently. We have a variable called expected containing the expected number of deaths from crude rates, and we are claiming `exposure(expected)`. What this is doing is saying that in two otherwise identical observations, if the number of expected deaths differed, we would expect the number of deaths due to melanoma to differ, too, and by the same proportion. See [SEM] `gsem family-and-link options`.

**Three-level Poisson model**

The same model, fit with Poisson, is
Example 39g — Three-level model (multilevel, generalized response)

To fit the model in the command language, we type

```
        . gsem (deaths <- uv M1[nation] M2[nation>region]), poisson exposure(expected)
```

Fitting fixed-effects model:

Iteration 0:  log likelihood = -2136.5847
Iteration 1:  log likelihood = -1723.8955
Iteration 2:  log likelihood = -1723.7727
Iteration 3:  log likelihood = -1723.7727
Refining starting values:

Grid node 0:  log likelihood = -1166.6536
Refining starting values (unscaled likelihoods):

Grid node 0:  log likelihood = -1166.6536
Fitting full model:

Iteration 0:  log likelihood = -1166.6536 (not concave)
Iteration 1:  log likelihood = -1152.2741 (not concave)
Iteration 2:  log likelihood = -1146.3094 (not concave)
Iteration 3:  log likelihood = -1119.8479 (not concave)
Iteration 4:  log likelihood = -1108.0129 (not concave)
Iteration 5:  log likelihood = -1098.8067
Iteration 6:  log likelihood = -1095.7563
Iteration 7:  log likelihood = -1095.3164
Iteration 8:  log likelihood = -1095.31
Iteration 9:  log likelihood = -1095.31

Generalized structural equation model

<table>
<thead>
<tr>
<th>Number of obs</th>
<th>354</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>deaths</td>
</tr>
<tr>
<td>Family</td>
<td>Poisson</td>
</tr>
<tr>
<td>Link</td>
<td>log</td>
</tr>
<tr>
<td>Log likelihood</td>
<td>-1095.31</td>
</tr>
</tbody>
</table>

( 1) \([\text{deaths}]M1[\text{nation}] = 1\)
( 2) \([\text{deaths}]M2[\text{nation}>\text{region}] = 1\)

|                | Coef.  | Std. Err. | z      | P>|z|    | [95% Conf. Interval] |
|----------------|--------|-----------|--------|--------|---------------------|
| deaths         |        |           |        |        |                     |
| uv             | -0.282041 | 0.0113998 | -2.47  | 0.013  | -0.0505473  -0.0058608 |
| M1[nation]     | 1 (constrained) |    |        |        |                     |
| M2[nation>region] | 1 (constrained) |    |        |        |                     |
| _cons          | -0.0639672 | 0.1335515 | -0.48  | 0.632  | -0.3257234  0.197789  |
| ln(expected)   | 1 (exposure) |    |        |        |                     |

|                | Coef.  | Std. Err. | z      | P>|z|    | [95% Conf. Interval] |
|----------------|--------|-----------|--------|--------|---------------------|
| var(M1[nation])| 0.1371732 | 0.0723303 |        |        | 0.048802  0.3855676 |
| var(M2[nation>region]) | 0.0483483 | 0.0109079 |        |        | 0.0310699  0.0752353 |

Testing for overdispersion

The negative binomial model allows for overdispersion, or in a multilevel framework, allows for conditional overdispersion. The Poisson model has no overdispersion, or in a multilevel model, no overdispersion beyond that predicted by the latent variables. We can test whether there is dispersion beyond what Poisson would predict:
We can reject at any reasonable level that the Poisson model adequately accounts for the dispersion in these data. Be aware that this test is conservative, because we are testing whether a variance goes to 0. \texttt{lrtest} usually issues a warning in such cases, but \texttt{lrtest} does not know that the relationship between negative binomial regression and Poisson regression involves a variance going to 0.

**Fitting the models with the Builder**

Use the diagram in *Three-level negative binomial model* above for reference.

1. Open the dataset.
   In the Command window, type
   
   \begin{verbatim}
   . use https://www.stata-press.com/data/r16/gsem_melanoma
   \end{verbatim}

2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in \texttt{gsem} mode by clicking on the $\displaystyle \mathcal{G}$ button.

4. Create the generalized response variable.
   a. Select the Add generalized response variable tool, $\square$.
   b. Click in the diagram about one-third of the way in from the right and one-fourth of the way up from the bottom.
   c. In the Contextual Toolbar, select \texttt{Nbinomial mean, Log} in the **Family/Link** control.
   d. In the Contextual Toolbar, select \texttt{deaths} in the **Variable** control.

5. Create the observed exogenous variable.
   a. Select the Add observed variable tool, $\square$, and then click in the diagram about one-third of the way in from the right and one-fourth of the way up from the bottom.
   b. In the Contextual Toolbar, select \texttt{uv} with the **Variable** control.

6. Create the level-three latent variable.
   a. Select the Add multilevel latent variable tool, $\ominus$, and click above the rectangle for \texttt{uv} about one-fourth of the way down from the top.
   b. In the Contextual Toolbar, click on the $\cdots$ button.
   c. Select the nesting level and nesting variable by selecting 2 from the **Nesting depth** control and selecting \texttt{nation > Observations} in the next line.
d. Specify M1 as the *Base name*.
e. Click on *OK*.

7. Create the level-two latent variable.
   a. Select the Add multilevel latent variable tool, ⤵, and click between the rectangle for *uv* and the double oval for *nation*.
   b. In the Contextual Toolbar, click on the 🔍 button.
   c. Select the nesting level and nesting variable by selecting 3 from the *Nesting depth* control and selecting *nation > region > Observations* in the next control.
   d. Specify M2 as the *Base name*.
e. Click on *OK*.

8. Create the paths from the exogenous variables to *deaths*.
   a. Select the Add path tool, ←.
   b. Click in the right side of the *uv* rectangle (it will highlight when you hover over it), and drag a path to the left side of the *deaths* rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the ← tool, draw paths from the right side of the double ovals for *nation* and *region* to the left side of the *deaths* rectangle.

9. Specify the level of exposure.
   Use the Select tool, ⍗, and double-click in the *deaths* rectangle. In the resulting dialog box, select *expected* in the *Exposure* control, and click on *OK*.

10. Clean up the location of the paths.
    If you do not like where the paths have been connected to the rectangles or oval, use the Select tool, ⍗, to click on the path, and then simply click on where it connects to a rectangle or oval and drag the endpoint.

11. Estimate.
    Click on the *Estimate* button, ⋅, in the Standard Toolbar, and then click on *OK* in the resulting *GSEM estimation options* dialog box.

    You can open a completed diagram in the Builder by typing 
    . webgetsem gsem_3lev

References


Also see

[SEM] **Example 38g** — Random-intercept and random-slope models (multilevel)

[SEM] **Example 34g** — Combined models (generalized responses)

[SEM] **Intro 5** — Tour of models

[SEM] **gsem** — Generalized structural equation model estimation command
Example 40g — Crossed models (multilevel)

Description

To illustrate crossed models, we use

```
. use https://www.stata-press.com/data/r16/gsem_fifeschool
(School data from Fife, Scotland)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_fifeschool.dta
obs: 3,435 School data from Fife, Scotland
vars: 5 25 Mar 2018 16:17
(_dta has notes)

<table>
<thead>
<tr>
<th>variable</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>int</td>
<td>%9.0g</td>
<td></td>
<td>Primary school ID</td>
</tr>
<tr>
<td>sid</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>Secondary school ID</td>
</tr>
<tr>
<td>attain</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>Attainment score at age 16</td>
</tr>
<tr>
<td>vrq</td>
<td>int</td>
<td>%9.0g</td>
<td></td>
<td>Verbal-reasoning score from final year of primary school</td>
</tr>
<tr>
<td>sex</td>
<td>byte</td>
<td>%9.0g</td>
<td></td>
<td>1: female; 0: male</td>
</tr>
</tbody>
</table>
```

Sorted by: pid sid

. notes

_dta:
2. Each observation is a different student. Each student attended a primary school (pid) and a secondary school (sid).
3. pid and sid are crossed, not nested. All combinations are possible.

Rabe-Hesketh and Skrondal (2012, 443–460) give an introduction to crossed-effects models and provide other examples of crossed-effects models by using the school data from Fife, Scotland.

See _Structural models 1: Linear regression_ and _Multilevel mixed-effects models_ in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- The crossed model
- Fitting the model with the Builder
The crossed model

In these data, 3,435 students attended 148 different primary schools and 19 different secondary schools. The model we wish to fit is

We include latent (random) effects for primary and secondary school because we think that school identities may have an effect.
The command syntax to fit this model is

```
.gsem (attain <- i.sex M1[pid] M2[sid])
```

Note: crossed random-effects model specified; option intmethod(laplace) implied

Fitting fixed-effects model:
Iteration 0:  log likelihood = -8701.1372
Iteration 1:  log likelihood = -8701.1372

Refining starting values:
Grid node 0:  log likelihood = -8587.4875

Fitting full model:
Iteration 0:  log likelihood = -8587.4875
Iteration 1:  log likelihood = -8573.5619
Iteration 2:  log likelihood = -8563.0147
Iteration 3:  log likelihood = -8561.7426
Iteration 4:  log likelihood = -8561.7354
Iteration 5:  log likelihood = -8561.7354

Generalized structural equation model

| Number of obs = 3,435 |

Response : attain
Family : Gaussian
Link : identity
Log likelihood = -8561.7354

( 1) [attain]M2[sid] = 1
( 2) [attain]M1[pid] = 1

| Coef. Std. Err.  z    P>|z|    [95% Conf. Interval] |
|------------------|---------------------|-------|-----------------------|
| attain           |                     |       |                       |
| 1.sex            | 0.4986152           | 0.0982634 | 5.07     | 0.000 | 0.3060224 | 0.6912079 |
| M1[pid]          | 1 (constrained)     |       |                       |
| M2[sid]          | 1 (constrained)     |       |                       |
| _cons            | 5.257372            | 0.1813785 | 28.99    | 0.000 | 4.901876 | 5.612867 |
| var(M1[pid])     | 1.104316            | 0.2022595 |           |       | 0.7712452 | 1.581226 |
| var(M2[sid])     | 0.3457088           | 0.160868 |           |       | 0.1388745 | 0.860594 |
| var(e.attain)    | 8.053437            | 0.1990023 |           |       | 7.672694 | 8.453074 |

Notes:
1. These data are not nested, but the diagram above would look the same even if they were. The fact that primary and secondary schools are crossed and not nested is, however, specified when we enter the model into the SEM Builder and is implicit in the command syntax.
2. We typed `attain <- i.sex M1[pid] M2[sid]`. We would have typed `attain <- i.sex M1[pid] M2[sid<pid]` had secondary school been nested within primary school.
3. `gsem` produced the following note when it began estimation: “crossed random effects detected; option intmethod(laplace) assumed”. `gsem` provides four integration methods. The default is `mvaghermite`, which stands for mean–variance adaptive Gauss–Hermite quadrature. The others are `mcaghermite` (mode-curvature adaptive Gauss–Hermite quadrature); `ghermite` (nonadaptive Gauss–Hermite quadrature); and `laplace` (Laplacian approximation).

In general, the adaptive methods `mvaghermite` and `mcaghermite` are considered superior in terms of accuracy to the nonadaptive method `ghermite`, which is considered superior to the approximation method `laplace`. They also take longer.
Fitting crossed models can be difficult. You may specify `intmethod()` with one of the superior methods, but be aware, convergence may not be achieved in finite time.

**Fitting the model with the Builder**

Use the diagram in *The crossed model* above for reference.

1. Open the dataset.

   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/gsem_fifeschool
   ```

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation.**

3. Put the Builder in `gsem` mode by clicking on the `g` button.

4. Create the endogenous variable.

   a. Select the Add observed variable tool, ``, and then click in the diagram about one-third of the way in from the right and halfway down from the top. After adding it, you can click inside the rectangle and move the variable if you wish.

   b. In the Contextual Toolbar, select `attain` with the `Variable` control.

5. Create the observed exogenous variable.

   a. Select the Add observed variable tool, ``, and then click in the diagram about one-third of the way in from the left and halfway down from the top.

   b. In the Contextual Toolbar, type `1.sex` in the `Variable` control (typing `1.sex` rather than using the `i` button to create `i.sex` prevents the rectangle corresponding to the base category for this binary variable from being created);

6. Create the `pid`-level latent variable.

   a. Select the Add multilevel latent variable tool, ``, and click below the rectangle for `attain`.

   b. In the Contextual Toolbar, click on the `` button.

   c. Select the nesting level and nesting variable by selecting 2 from the `Nesting depth` control and selecting `pid > Observations` in the next control.

   d. Specify `M1` as the `Base name`.

   e. Click on `OK`.

7. Create the `sid`-level latent variable.

   a. Select the Add multilevel latent variable tool, ``, and click above the rectangle for `attain`.

   b. In the Contextual Toolbar, click on the `` button.

   c. Select the nesting level and nesting variable by selecting 2 from the `Nesting depth` control and selecting `sid > Observations` in the next control.

   d. Specify `M2` as the `Base name`.

   e. Click on `OK`. 
8. Create the paths from the exogenous variables to \textit{attain}.
   \begin{itemize}
   \item[a.] Select the Add path tool, \textit{-}.
   \item[b.] Click in the right side of the \textit{1.sex} rectangle (it will highlight when you hover over it), and drag a path to the left side of the \textit{attain} rectangle (it will highlight when you can release to connect the path).
   \item[c.] Continuing with the \textit{-} tool, draw paths from bottom of the \textit{sid}_2 double oval to the top of the \textit{attain} rectangle and from the top of the \textit{pid}_1 double oval to the bottom of the \textit{attain} rectangle.
   \end{itemize}

9. Clean up the location of the paths.
   If you do not like where the paths have been connected to the rectangles or oval, use the Select tool, \textit{\textcolor{red}{A}}, to click on the path, and then simply click on where it connects to a rectangle or oval and drag the endpoint.

10. Estimate.
    Click on the \textbf{Estimate} button, \textit{\textcolor{green}{E}}, in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{GSEM estimation options} dialog box.

    You can open a completed diagram in the Builder by typing
    \begin{verbatim}
    . webgetsem gsem_cross
    \end{verbatim}

\section*{Reference}

\section*{Also see}
\textbf{SEM Example 38g} — Random-intercept and random-slope models (multilevel)
\textbf{SEM Intro 5} — Tour of models
\textbf{SEM gsem} — Generalized structural equation model estimation command
Example 41g — Two-level multinomial logistic regression (multilevel)

Description

We demonstrate two-level multinomial logistic regression with random effects by using the following data:

```stata
use https://www.stata-press.com/data/r16/gsem_lineup
(Fictional suspect identification data)
describe
Contains data from https://www.stata-press.com/data/r16/gsem_lineup.dta
obs: 6,535 Fictional suspect identification data
vars: 6 29 Mar 2018 10:35
(_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage type</th>
<th>display format</th>
<th>value label</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>suspect</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>suspect id</td>
</tr>
<tr>
<td>suswhite</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>suspect is white</td>
</tr>
<tr>
<td>violent</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>violent crime</td>
</tr>
<tr>
<td>location</td>
<td>float</td>
<td>%14.0g</td>
<td>loc</td>
<td>lineup location</td>
</tr>
<tr>
<td>witmale</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>witness is male</td>
</tr>
<tr>
<td>chosen</td>
<td>float</td>
<td>%9.0g</td>
<td>choice</td>
<td>individual identified in lineup by witness</td>
</tr>
</tbody>
</table>

Sorted by: suspect
describe
_dta:
1. Fictional data inspired by Wright, D.B and Sparks, A.T., 1994, "Using multilevel multinomial regression to analyse line-up data", _Multilevel Modeling Newsletter_, Vol. 6, No. 1
2. Data contain repeated values of variable suspect. Each suspect is viewed by multiple witnesses and each witness (1) declines to identify a suspect, (2) chooses a foil, or (3) chooses the suspect.

tabulate location

<table>
<thead>
<tr>
<th>lineup location</th>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>police_station</td>
<td>2,228</td>
<td>34.09</td>
<td>34.09</td>
</tr>
<tr>
<td>suite_1</td>
<td>1,845</td>
<td>28.23</td>
<td>62.33</td>
</tr>
<tr>
<td>suite_2</td>
<td>2,462</td>
<td>37.67</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Total 6,535 100.00
In what follows, we re-create results similar to those of Wright and Sparks (1994), but we use fictional data. These data resemble the real data used by the authors in proportion of observations having each level of the outcome variable chosen, and the data produce results similar to those presented by the authors.

See Structural models 6: Multinomial logistic regression and Multilevel mixed-effects models in [SEM] Intro 5 for background.

For additional discussion of fitting multilevel multinomial logistic regression models, see Skrondal and Rabe-Hesketh (2003).

Remarks and examples

Remarks are presented under the following headings:

Two-level multinomial logistic model with shared random effects
Two-level multinomial logistic model with separate but correlated random effects
Fitting the model with the Builder

Two-level multinomial logistic model with shared random effects

We wish to fit the following model:
This model concerns who is chosen in a police lineup. The response variables are 1.chosen, 2.chosen, and 3.chosen, meaning chosen = 1 (code for not chosen), chosen = 2 (code for foil chosen), and chosen = 3 (code for suspect chosen). A foil is a stand-in who could not possibly be guilty of the crime.

We say the response variables are 1.chosen, 2.chosen, and 3.chosen, but 1.chosen does not even appear in the diagram. By its omission, we are specifying that chosen = 1 be treated as the base mlogit category. There are other ways we could have drawn this; see [SEM] Example 37g.

In these data, each suspect was viewed by multiple witnesses. In the model, we include a random effect at the suspect level, and we constrain the effect to be equal for chosen values 2 and 3 (selecting the foil or the suspect).

We can fit this model with command syntax by typing
. gsem (i.chosen <- i.location i.suswhite i.witmale i.violent M1[suspect]@1), > mlogit

Fitting fixed-effects model:
Iteration 0: log likelihood = -6914.9098
Iteration 1: log likelihood = -6696.7136
Iteration 2: log likelihood = -6694.0006
Iteration 3: log likelihood = -6693.9974
Iteration 4: log likelihood = -6693.9974

Refining starting values:
Grid node 0: log likelihood = -6705.0919

Fitting full model:
Iteration 0: log likelihood = -6705.0919 (not concave)
Iteration 1: log likelihood = -6654.5724
Iteration 2: log likelihood = -6653.5717
Iteration 3: log likelihood = -6653.5671
Iteration 4: log likelihood = -6653.5671

Generalized structural equation model
Response : chosen
Base outcome : 1
Family : multinomial
Link : logit
Log likelihood = -6653.5671

( 1) [2.chosen]M1[suspect] = 1
( 2) [3.chosen]M1[suspect] = 1

|                      | Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|----------------------|-------|-----------|-------|-------|---------------------|
| 1.chosen             |       |           |       |       |                     |
| (base outcome)       |       |           |       |       |                     |
| 2.chosen             |       |           |       |       |                     |
| location             |       |           |       |       |                     |
| suite_1              | .3867 | .1027     | 3.76  | 0.000 | .1853868 .5880264  |
| suite_2              | .4916 | .0980     | 5.01  | 0.000 | .2994299 .6837051  |
| 1.suswhite           | -.0275| .0751     | -0.37 | 0.714 | -.1748736 .1197734 |
| 1.witmale            | -.0001| .0681     | -0.00 | 0.998 | -.1336193 .1332505 |
| 1.violent           | .0356 | .0774     | 0.46  | 0.645 | -.1159864 .1872819 |
| M1[suspect]          |       |           |       |       |                     |
| _cons                | -1.00 | .0993     | -10.09| 0.000 | -1.197003 -.8076643 |
| 3.chosen             |       |           |       |       |                     |
| location             |       |           |       |       |                     |
| suite_1              | -.2832| .0936     | -3.02 | 0.002 | -.4667271 -.0996814 |
| suite_2              | .1392 | .0863     | 1.61  | 0.107 | -.0300581 .3084172 |
| 1.suswhite           | -.2398| .0643     | -3.73 | 0.000 | -.3657965 -.1137158 |
| 1.witmale            | .1419 | .0593     | 2.39  | 0.017 | .0256712 .2581857  |
| 1.violent           | -1.38 | .0885     | -15.55| 0.000 | -1.55006 -1.203097 |
| M1[suspect]          |       |           |       |       |                     |
| _cons                | .1781 | .0834     | 2.14  | 0.033 | .0147627 .3414468  |
| var( M1[suspect])   | .2538 | .0427     | 5.83  | 0.000 | .1824673 .3530228  |
Notes:

1. We show the interpretation of mlogit coefficients in [SEM] Example 37g.

2. The estimated variance of the random effect is 0.2538, implying a standard deviation of 0.5038. Thus a 1-standard-deviation change in the random effect amounts to a \( \exp(0.5038) = 1.655 \) change in the relative-risk ratio. The effect is both practically significant and, from the output, statistically significant.

3. This is not the model fit by Wright and Sparks (1994). Those authors did not constrain the random effect to be the same for \texttt{chosen} equal to 2 and 3. They included separate but correlated random effects, and then took that even a step further.

Two-level multinomial logistic model with separate but correlated random effects

The model we wish to fit is

\[
\begin{align*}
\text{1b.location} \\
\text{2.location} \\
\text{3.location} \\
\text{1.suswhite} \\
\text{1.witmale} \\
\text{1.violent} \\
\end{align*}
\]

This is one of the models fit by Wright and Sparks (1994), although remember that we are using fictional data.

We can fit this model with command syntax by typing

\[
\text{. gsem (2.chosen <- i.location i.suswhite i.witmale i.violent M1[suspect])} \quad ///
\]
\[> \quad (3.chosen <- i.location i.suswhite i.witmale i.violent M2[suspect]), \quad ///
\]
\[> \quad \text{mlogit}
\]

We did not even mention the assumed covariance between the random effects because latent exogenous variables are assumed to be correlated in the command language. Even so, we can specify the \texttt{cov()} option if we wish, and we might do so for emphasis or because we are unsure whether the parameter would be included.
. gsem (2.chosen <- i.location i.suswhite i.witmale i.violent M1[suspect])
> (3.chosen <- i.location i.suswhite i.witmale i.violent M2[suspect]),
>     cov(M1[suspect]*M2[suspect]) mlogit

Fitting fixed-effects model:
Iteration 0:  log likelihood =  -6914.9098
Iteration 1:  log likelihood = -6696.7136
Iteration 2:  log likelihood = -6694.0006
Iteration 3:  log likelihood = -6693.9974
Iteration 4:  log likelihood = -6693.9974

Refining starting values:
Grid node 0:  log likelihood =  -6793.4228

Fitting full model:
Iteration 0:  log likelihood =  -6793.4228  (not concave)
Iteration 1:  log likelihood = -6717.7507  (not concave)
Iteration 2:  log likelihood = -6684.6592
Iteration 3:  log likelihood = -6660.1404
Iteration 4:  log likelihood = -6652.1368
Iteration 5:  log likelihood = -6651.7841
Iteration 6:  log likelihood = -6651.7819
Iteration 7:  log likelihood = -6651.7819

Generalized structural equation model  Number of obs  =  6,535
Response : chosen
Base outcome : 1
Family : multinomial
Link : logit
Log likelihood = -6651.7819

( 1) [2.chosen]M1[suspect] = 1
( 2) [3.chosen]M2[suspect] = 1

| Coef.  | Std. Err. |     z  |   P>|z|  | [95% Conf. Interval] |
|--------|-----------|--------|--------|----------------------|
| 1.chosen |          | (base outcome) |
| 2.chosen | location |         |        |                      |
| suite_1     |     0.3881676 | 0.1004754 | 3.86   | 0.000               | 0.1912394 | 0.5850958 |
| suite_2     |     0.48938   | 0.0960311 | 5.10   | 0.000               | 0.3011625 | 0.6775974 |
| 1.suswhite   | -0.0260152   | 0.0749378 | -0.35  | 0.728               | -0.1728906 | 0.1208602 |
| 1.witmale    | -0.0007652   | 0.0679187 | -0.01  | 0.991               | -0.1338833 | 0.132353  |
| 1.violent    | 0.0369381    | 0.0771594 | 0.48   | 0.632               | -0.1142915 | 0.1881677 |

M1[suspect] 1 (constrained)

_cons        | -1.000382    | 0.0992546  | -10.08 | 0.000               | -1.194918  | -0.8058469 |

3.chosen | location |         |        |                      |
| suite_1     |   -0.2904225 | 0.0685785 | -3.00  | 0.003               | -0.4802604 | -0.1005847 |
| suite_2     |   0.1364246  | 0.0892821 | 1.53   | 0.127               | -0.0385649 | 0.3114142  |
| 1.suswhite   | -0.2437654   | 0.0647275 | -3.77  | 0.000               | -0.370629  | -0.1169018 |
| 1.witmale    | 0.139826     | 0.0596884 | 2.34   | 0.019               | 0.0228389  | 0.256813   |
| 1.violent    | -1.388013    | 0.0891863 | -15.56 | 0.000               | -1.562815  | -1.213212  |

M2[suspect] 1 (constrained)

_cons        | 0.1750622    | 0.0851614  | 2.06   | 0.040               | 0.008149   | 0.3419754  |
Notes:

1. The estimated variances of the two random effects are 0.2168 and 0.2978, which as explained in the second note of above example, are both practically and statistically significant.

2. The covariance is estimated to be 0.2300. Therefore, \( \frac{0.2300}{\sqrt{0.2168 \times 0.2978}} = 0.9052 \) is the estimated correlation.

3. Wright and Sparks (1994) were interested in whether the location of the lineup mattered. They found that it did, and that foils were more likely to be chosen at lineups outside of the police station (at the two “specialist” suites). They speculated the cause might be that the police at the station strongly warn witnesses against misidentification, or possibly because the specialist suites had better foils.

### Fitting the model with the Builder

Use the first diagram in *Two-level multinomial logistic model with shared random effects* above for reference.

1. Open the dataset.

   In the Command window, type:
   ```stata
   . use https://www.stata-press.com/data/r16/gsem_lineup
   ```

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in *gsem* mode by clicking on the \( \text{ setContentView} \) button.

4. Create the independent variables.

   Select the Add observed variables set tool, \( \text{Add to varlist} \), and then click near the bottom of the diagram about one-third of the way in from the left.

   In the resulting dialog box,
   a. select the *Select variables* radio button (it may already be selected);
   b. include the levels of the factor variable *location* by clicking on the """" button next to the *Variables* control. In the resulting dialog box, select the *Factor variable* radio button, select *Main effect* in the *Specification* control, and select *location* in the *Variables* control for *Variable 1*. Click on *Add to varlist*, and then click on *OK*;
   c. type `1.suswhite 1.witmale 1.violent` in the *Variables* control after `i.location` (typing `1.varname` rather than using the """" button to create them as `i.varname` factor variables prevents rectangles corresponding to the base categories for these binary variables from being created);
d. select Vertical in the Orientation control;
e. click on OK.

If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the rectangles for the possible outcomes of the multinomial endogenous variable.

Select the Add observed variables set tool, \( \cdots \), and then click in the diagram about one-third of the way in from the right and one-fourth of the way up from the bottom.

In the resulting dialog box,

a. select the Select variables radio button (it may already be selected);
b. check Make variables generalized responses;
c. select Multinomial, Logit in the Family/Link control;
d. select chosen in the Variable control;
e. under Levels, remove 1b to prevent the rectangle corresponding to the base category from being created;
f. select Vertical in the Orientation control;
g. select the Distances tab;
h. select .5 (inch) from the from the Distance between variables control;
i. click on OK.

6. Create the paths from the independent variables to the rectangles for outcomes chosen = 2 and chosen = 3.

a. Select the Add path tool, \( \leftarrow \).
b. Click in the right side of the 1b.location rectangle (it will highlight when you hover over it), and drag a path to the left side of the 2.chosen rectangle (it will highlight when you can release to connect the path).
c. Continuing with the \( \leftarrow \) tool, click in the right side of each independent variable and drag a path to both the 2.chosen and 3.chosen rectangles.

7. Create the suspect-level latent variable.

a. Select the Add multilevel latent variable tool, \( \circ \), and click near the right side of the diagram, vertically centered between 2.chosen and 3.chosen.
b. In the Contextual Toolbar, click on the \( \cdots \) button.
c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting suspect > Observations in the next control.
d. Specify M1 as the Base name.
e. Click on OK.

8. Create the paths from the multilevel latent variable to the rectangles for outcomes chosen = 2 and chosen = 3.

a. Select the Add path tool, \( \leftarrow \).
b. Click in the upper-left quadrant of the suspect1 double oval, and drag a path to the right side of the 2.chosen rectangle.
c. Continuing with the tool, click in the lower-left quadrant of the suspect₁ double oval, and drag a path to the right side of the 3.chosen rectangle.

9. Place constraints on path coefficients from the multilevel latent variable.

Use the Select tool, to select the path from the suspect₁ double oval to the 2.chosen rectangle. Type 1 in the # box in the Contextual Toolbar and press Enter. Repeat this process to constrain the coefficient on the path from the suspect₁ double oval to the 3.chosen rectangle to 1.

10. Clean up the location of the paths.

If you do not like where the paths have been connected to the rectangles, use the Select tool, to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.

11. Estimate.

Click on the Estimate button, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

12. If you wish to fit the model described in Two-level multinomial logistic model with separate but correlated random effects, use the Select tool to select the path from the suspect₁ double oval to the 3.chosen rectangle in the diagram created above. Select Object > Delete from the SEM Builder menu.

Using the Select tool, select the suspect₁ double oval and move it up so that it is parallel with the rectangle for 2.chosen.

13. Create the multilevel latent variable corresponding to the random effects of suspect in the 3.chosen equation.

a. Select the Add multilevel latent variable tool, and click near the right side of the diagram, next to the 3.chosen rectangle.

b. In the Contextual Toolbar, click on the button.

c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting suspect > Observations in the next control.

d. Specify M2 as the Base name.

e. Click on OK.

14. Draw a path from the newly added suspect-level latent variable to 3.chosen.

Select the Add path tool, click in the left of the suspect₂ double oval, and drag a path to the right side of the 3.chosen rectangle.

15. Create the covariance between the random effects.

a. Select the Add covariance tool, .

b. Click in the bottom-right quadrant of the suspect₁ double oval, and drag a covariance to the top right of the suspect₂ double oval.

16. Clean up paths and covariance.

If you do not like where a path has been connected to its variables, use the Select tool, , to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint. Similarly, you can change where the covariance connects to the latent variables by clicking on the covariance and dragging the endpoint. You can also change the bow of the covariance by
clicking on the covariance and dragging the control point that extends from one end of the selected covariance.

17. Estimate again.

Click on the **Estimate** button, ![Estimate](image), in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

You can open a completed diagram for the first model in the Builder by typing

```
.webgetsem gsem_mlmlogit1
```

You can open a completed diagram for the second model in the Builder by typing

```
.webgetsem gsem_mlmlogit2
```

References


Also see

[SEM] **Example 37g** — Multinomial logistic regression

[SEM] **Example 38g** — Random-intercept and random-slope models (multilevel)

[SEM] **Intro 5** — Tour of models

[SEM] **gsem** — Generalized structural equation model estimation command
To demonstrate linear mediation models, we use the following data:

```stata
describe

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>branch</td>
<td>1,500</td>
<td>38</td>
<td>21.65593</td>
<td>1</td>
<td>75</td>
</tr>
<tr>
<td>support</td>
<td>1,500</td>
<td>.0084667</td>
<td>.5058316</td>
<td>-1.6</td>
<td>1.8</td>
</tr>
<tr>
<td>satis</td>
<td>1,500</td>
<td>.0212</td>
<td>.6087235</td>
<td>-1.6</td>
<td>2</td>
</tr>
<tr>
<td>perform</td>
<td>1,500</td>
<td>5.005317</td>
<td>.8949845</td>
<td>2.35022</td>
<td>8.084294</td>
</tr>
</tbody>
</table>
```

Notes:
1. Fictional data on job performance, job satisfaction, and perceived support from managers for 1,500 sales employees of a large department store in 75 locations.
2. Variable support is average of Likert-scale questions, each question scored from -2 to 2.
3. Variable satis is average of Likert-scale questions, each question scored from -2 to 2.
4. Variable perform is job performance measured on continuous scale.

See *Structural models 1: Linear regression* and *Multilevel mixed-effects models* in [SEM] *Intro 5* for background.
The model we wish to fit is the simplest form of a mediation model, namely,

![Diagram]

We are interested in the effect of managerial support on job performance, but we suspect a portion of the effect might be mediated through job satisfaction. In traditional mediation analysis, the model would be fit by a series of linear regression models as described in Baron and Kenny (1986). That approach is sufficient because the errors are not correlated. The advantage of using structural equation modeling is that you can fit a single model and estimate the indirect and total effects, and you can embed the simple mediation model in a larger model and even use latent variables to measure any piece of the mediation model.

To fit this model with the command syntax, we type

```
. sem (perform <- satis support) (satis <- support)
```

Endogenous variables
Observed: perform satis
Exogenous variables
Observed: support

Fitting target model:
Iteration 0: log likelihood = -3779.9224
Iteration 1: log likelihood = -3779.9224

Structural equation model  Number of obs = 1,500
Estimation method = ml
Log likelihood = -3779.9224

| QIM | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|-----|-------|-----------|---|---------|---------------------|
| Structural perform | | | | | | |
| satis | .8984401 | .0251903 | 35.67 | 0.000 | .849068 | .9478123 |
| support | .6161077 | .0303143 | 20.32 | 0.000 | .5566927 | .6755227 |
| _cons | 4.981054 | .0150589 | 330.77 | 0.000 | 4.951539 | 5.010569 |
| satis support | | | | | | |
| support | .2288945 | .0305047 | 7.50 | 0.000 | .1691064 | .2886826 |
| _cons | .019262 | .0154273 | 1.25 | 0.212 | -.0109749 | .0494989 |
| var(e.perform) | .3397087 | .0124044 | | | .3162461 | .364912 |
| var(e.satis) | .3569007 | .0130322 | | | .3322507 | .3833795 |

LR test of model vs. saturated: chi2(0) = 0.00, Prob > chi2 = .
Notes:

1. The direct effect of managerial support on job performance is measured by `perform <- support` and is estimated to be 0.6161. The effect is small albeit highly statistically significant. The standard deviations of performance and support are 0.89 and 0.51. A one standard deviation increase in support improves performance by a third of a standard deviation.

2. The direct effect of job satisfaction on job performance is measured by `perform <- satis` and is estimated to be 0.8984. That also is a moderate effect, practically speaking, and is highly statistically significant.

3. The effect of managerial support on job satisfaction is measured by `satis <- support` and is practically small but statistically significant.

4. What is the total effect of managerial support on performance? It is the direct effect (0.6161) plus the indirect effect of support on satisfaction on performance \((0.2289 \times 0.8984 = 0.2056)\), meaning the total effect is 0.8217. It would be desirable to put a standard error on that, but that’s more work.

We can use `estat teffects` after estimation to obtain the total effect and its standard error:

\[
\begin{align*}
\text{. estat teffects} \\
\text{Direct effects} \\
\begin{array}{cccccc}
\text{Structural} & \text{Coef.} & \text{Std. Err.} & z & P>|z| & \text{[95% Conf. Interval]} \\
\text{perform} & .8984401 & .0251903 & 35.67 & 0.000 & .849068 & .9478123 \\
satis & .6161077 & .0303143 & 20.32 & 0.000 & .5566927 & .6755227 \\
satis & .2288945 & .0305047 & 7.50 & 0.000 & .1691064 & .2886826 \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{Indirect effects} \\
\begin{array}{cccccc}
\text{Structural} & \text{Coef.} & \text{Std. Err.} & z & P>|z| & \text{[95% Conf. Interval]} \\
\text{perform} & .205648 & .0280066 & 7.34 & 0.000 & .150756 & .26054 \\
satis & 0 (no path) \\
satis & 0 (no path) \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{Total effects} \\
\begin{array}{cccccc}
\text{Structural} & \text{Coef.} & \text{Std. Err.} & z & P>|z| & \text{[95% Conf. Interval]} \\
\text{perform} & .8984401 & .0251903 & 35.67 & 0.000 & .849068 & .9478123 \\
satis & .8217557 & .0404579 & 20.31 & 0.000 & .7424597 & .9010516 \\
satis & .2288945 & .0305047 & 7.50 & 0.000 & .1691064 & .2886826 \\
\end{array}
\end{align*}
\]
One-level model with \texttt{gsem}

We can fit the same model with \texttt{gsem}. The command is the same except that we substitute \texttt{gsem} for \texttt{sem}, and results are identical:

\begin{verbatim}
. gsem (perform <- satis support) (satis <- support)
Iteration 0:  log likelihood = -2674.3421
Iteration 1:  log likelihood = -2674.3421  (backed up)
Generalized structural equation model       Number of obs  =  1,500
Response : perform
Family    : Gaussian
Link      : identity
Response  : satis
Family    : Gaussian
Link      : identity
Log likelihood = -2674.3421

|                      | Coef.  | Std. Err. |      z    |   P>|z|   |     [95% Conf. Interval]      |
|----------------------|--------|-----------|-----------|-------|-------------------------------|
| perform              |        |           |           |       |                               |
| satis                | .8984401 | .0251903  | 35.67     | 0.000 | .849068                      | .9478123    |
| support              | .6161077 | .0303143  | 20.32     | 0.000 | .5566927                     | .6755227    |
| _cons                | 4.981054 | .0150589  | 330.77    | 0.000 | 4.951539                      | 5.010569    |
| satis                |        |           |           |       |                               |
| support              | .2288945 | .0305047  | 7.50      | 0.000 | .1691064                     | .2886826    |
| _cons                | .019262  | .0154273  | 1.25      | 0.212 | -.0109749                    | .0494989    |
| var(e.perform)       | .3397087 | .0124044  |           |       | .3162461                     | .364912     |
| var(e.satis)         | .3569007 | .0130322  |           |       | .3322507                     | .3833795    |
\end{verbatim}

After \texttt{gsem}, however, we cannot use \texttt{estat teffects}:

\begin{verbatim}
. estat teffects
  estat teffects not valid
r(321);
\end{verbatim}

We can, however, calculate the indirect and total effects for ourselves and obtain the standard error by using \texttt{nlcom}. Referring back to note 4 of the previous section, the formula for the indirect effect and total effects are

\[
\text{indirect effect} = \beta_1 \beta_4 \\
\text{total effect} = \beta_2 + \beta_1 \beta_4
\]

where

\[
\beta_1 = \text{path coefficient for } \text{perform} \leftarrow \text{satis} \\
\beta_4 = \text{path coefficient for } \text{satis} \leftarrow \text{support} \\
\beta_2 = \text{path coefficient for } \text{perform} \leftarrow \text{support}
\]

It turns out that we can access the coefficients by typing

\[
\beta_1 = \_b[\text{perform:satis}] \\
\beta_4 = \_b[\text{satis:support}] \\
\beta_2 = \_b[\text{perform:support}]
\]
which is most easily revealed by typing

```
    . gsem, coeflegend
    (output omitted)
```

Thus we can obtain the indirect effect by typing

```
    . nlcom _b[perform:satis]*_b[satis:support]
    _nl_1: _b[perform:satis]*_b[satis:support]
```

|          | Coef. | Std. Err. | z     | P>|z|  | 95% Conf. Interval |
|----------|-------|-----------|-------|------|--------------------|
| _nl_1    | .205648 | .0280066  | 7.34  | 0.000 | .150756 .26054     |

and we can obtain the total effect by typing

```
    . nlcom _b[perform:support]+_b[perform:satis]*_b[satis:support]
    _nl_1: _b[perform:support]+_b[perform:satis]*_b[satis:support]
```

|          | Coef. | Std. Err. | z     | P>|z|  | 95% Conf. Interval |
|----------|-------|-----------|-------|------|--------------------|
| _nl_1    | .8217557 | .0404579  | 20.31 | 0.000 | .7424597 .9010516 |

**Two-level model with **gsem**

It may be easier to use **sem** rather than **gsem** for fitting single-level models, but if you want to fit multilevel models, you must use **gsem**. A variation on the model we just fit is

![Diagram of two-level model](image)

In this model, we include a random intercept in each equation at the branch (individual store) level. The model above is one of many variations on two-level mediation models; see Krull and MacKinnon (2001) for an introduction to multilevel mediation models, and see Preacher, Zyphur, and Zhang (2010) for a discussion of fitting these models with structural equation modeling.
To fit this model with the command syntax, we type

```stata
.gsem (perform <- satis support M1[branch]) (satis <- support M2[branch]),
> cov(M1[branch]*M2[branch])*0
```

Fitting fixed-effects model:

Iteration 0: log likelihood = -2674.3421
Iteration 1: log likelihood = -2674.3421

Refining starting values:

Grid node 0: log likelihood = -2132.1613

Fitting full model:

Iteration 0: log likelihood = -2132.1613 (not concave)
Iteration 1: log likelihood = -1801.3155
Iteration 2: log likelihood = -1769.6421
Iteration 3: log likelihood = -1705.1282
Iteration 4: log likelihood = -1703.746
Iteration 5: log likelihood = -1703.7141
Iteration 6: log likelihood = -1703.714

Generalized structural equation model

Number of obs = 1,500
Response : perform
Family : Gaussian
Link : identity
Response : satis
Family : Gaussian
Link : identity

Log likelihood = -1703.714

( 1) [perform]M1[branch] = 1
( 2) [satis]M2[branch] = 1

|              | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------------|--------|-----------|-------|------|----------------------|
| perform      |        |           |       |      |                      |
| satis        | .604264| .0336398  | 17.96 | 0.000| .5383313 .6701968   |
| support      | .6981525| .0250432  | 27.88 | 0.000| .6490687 .7472364 |
| M1[branch]   | 1      | (constrained) |     |      |                      |
| _cons        | 4.986596| .0489465  | 101.88| 0.000| 4.890663 .5082529 |
| satis        |        |           |       |      |                      |
| support      | .2692633| .0179649  | 14.99 | 0.000| .2340528 .3044739 |
| M2[branch]   | 1      | (constrained) |     |      |                      |
| _cons        | .0189202| .0570868  | 0.33  | 0.740| -.0929678 .1308083 |
| var(M1[branch]) | .1695962| .0302866  |       |      | .119511 .2406713 |
| var(M2[branch]) | .2384738| .0399154  |       |      | .1717781 .3310652 |
| var(e.perform) | .201053| .0075451  |       |      | .1867957 .2163985 |
| var(e.satis)  | .1188436| .0044523  |       |      | .1104299 .1278983 |

Notes:

1. In One-level model with sem above, we measured the direct effects on job performance of job satisfaction and managerial support as 0.8984 and 0.6161. Now the direct effects are 0.6043 and 0.6982.
2. We can calculate the indirect and total effects just as we did in the previous section, which we will do below. We mentioned earlier that there are other variations of two-level mediation models, and how you calculate total effects depends on the model chosen.

In this case, the indirect effect is

\[
\text{. \texttt{nlcom} } \_b\{\text{perform:satis}\}*\_b\{\text{satis:support}\}
\]

\[
\text{ \_\texttt{nl1}: } \_b\{\text{perform:satis}\}*\_b\{\text{satis:support}\}
\]

|             | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------------|-------|-----------|------|------|---------------------|
| _\texttt{nl1}_ | 0.1627062 | 0.0141382 | 11.51 | 0.000 | 0.1349958 to 0.1904165 |

and the total effect is

\[
\text{. \texttt{nlcom} } \_b\{\text{perform:support}\}+\_b\{\text{perform:satis}\}*\_b\{\text{satis:support}\}
\]

\[
\text{ \_\texttt{nl1}: } \_b\{\text{perform:support}\}+\_b\{\text{perform:satis}\}*\_b\{\text{satis:support}\}
\]

|             | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------------|-------|-----------|------|------|---------------------|
| _\texttt{nl1}_ | 0.8608587 | 0.0257501 | 33.43 | 0.000 | 0.8103894 to 0.911328 |

### Fitting the models with the Builder

Use the diagram in *One-level model with sem* above for reference.

1. Open the dataset.

   In the Command window, type

   \[
   \text{. use https://www.stata-press.com/data/r16/gsem_multmed}
   \]

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Create a regression component for the perform outcome.

   Select the Add regression component tool, \(\text{\textbullet}\), and then click in the center of the diagram.

   In the resulting dialog box,

   a. select *perform* in the *Dependent variable* control;
   
   b. select *support* with the *Independent variables* control;
   
   c. select *Left* in the *Independent variables’ direction* control;
   
   d. click on *OK*.

   e. Use the Select tool, \(\text{\textbullet}\), to select only the *perform* rectangle, and drag it to the right to increase the distance between the rectangles. (You can hold the *Shift* key while dragging to ensure that the movement is directly to the right.)

4. Create the mediating variable.

   a. Select the Add observed variable tool, \(\text{\square}\), and then click in the diagram above the path from *support* to *perform*.

   b. In the Contextual Toolbar, select *satis* with the *Variable* control.
5. Create the paths to and from the mediating variable.
   a. Select the Add path tool, \( \rightarrow \).
   b. Click in the upper right of the support rectangle (it will highlight when you hover over it), and drag a path to the lower left of the satisf rectangle (it will highlight when you can release to connect the path).
   c. Continuing with the \( \rightarrow \) tool, draw a path from the lower right of the satisf rectangle to the upper left of the perform rectangle.

6. Clean up the direction of the error term.
   We want the error for each of the endogenous variables to be to the right of the rectangle. The error for satisf may have been created in another direction. If so,
   a. choose the Select tool, \( \text{select tool} \);
   b. click in the satisf rectangle;
   c. click on one of the Error rotation buttons, \( \text{rotation} \), in the Contextual Toolbar until the error is to the right of the rectangle.

7. Clean up the location of the paths.
   If you do not like where the paths have been connected to the rectangles, use the Select tool, \( \text{select tool} \), to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.

8. Estimate.
   Click on the Estimate button, \( \text{estimate} \), in the Standard Toolbar, and then click on OK in the resulting SEM estimation options dialog box.

9. To fit the model in Two-level model with gsem, continue with the previous diagram, and put the builder in gsem mode by clicking on the \( \text{gsem} \) button.

10. Create the multilevel latent variable corresponding to the random intercept for satisf.
    a. Select the Add multilevel latent variable tool, \( \text{latent variable} \), and click above the rectangle for satisf.
    b. In the Contextual Toolbar, click on the \( \text{nesting} \) button.
    c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting branch > Observations in the next control.
    d. Specify M1 as the Base name.
    e. Click on OK.

11. Create the multilevel latent variable corresponding to the random intercept for perform.
    a. Select the Add multilevel latent variable tool, \( \text{latent variable} \), and click above the rectangle for satisf and to the right of the branch1 double oval.
    b. In the Contextual Toolbar, click on the \( \text{nesting} \) button.
    c. Select the nesting level and nesting variable by selecting 2 from the Nesting depth control and selecting branch > Observations in the next control.
    d. Specify M2 as the Base name.
    e. Click on OK.
12. Draw paths from the multilevel latent variables to their corresponding endogenous variables.
   a. Select the Add path tool, \(^{-}\).
   b. Click in the bottom of the \(\text{branch}_1\) double oval, and drag a path to the top of the \(\text{satis}\) rectangle.
   c. Continuing with the \(^{-}\) tool, click in the bottom of the \(\text{branch}_2\) double oval, and drag a path to the top of the \(\text{perform}\) rectangle.

13. Estimate again.
   Click on the \textbf{Estimate} button, \(^{\text{•}}\), in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{GSEM estimation options} dialog box.

You can open a completed diagram for the first model in the Builder by typing
\[\text{. webgetsem sem_med}\]
You can open a completed diagram for the second model in the Builder by typing
\[\text{. webgetsem gsem_mlmed}\]

References

Also see
[SEM] \textbf{Example 38g} — Random-intercept and random-slope models (multilevel)
[SEM] \textbf{Intro 5} — Tour of models
[SEM] \textbf{gsem} — Generalized structural equation model estimation command
Tobit regression is demonstrated using auto.dta:

```
. use https://www.stata-press.com/data/r16/auto
    (1978 Automobile Data)
. generate wgt = weight/1000
```

See *Structural models 1: Linear regression* in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting tobit regression models
- Fitting the model with the Builder

Fitting tobit regression models

The first example in [R] tobit is

```
. tobit mpg wgt, ll(17)
```

This model corresponds to

Censoring information does not appear in the path diagram by default. It can be added to the path diagram by customizing the appearance of mpg in the Builder. The Builder reports the censoring information for mpg in the Details pane.
To fit this model with \texttt{gsem}, we type

\begin{verbatim}
. gsem mpg <- wgt, family(gaussian, lencoded(17))
\end{verbatim}

Refining starting values:

Grid node 0: log likelihood = -170.32555

Fitting full model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-170.32555</td>
</tr>
<tr>
<td>1</td>
<td>-164.66209</td>
</tr>
<tr>
<td>2</td>
<td>-164.25471</td>
</tr>
<tr>
<td>3</td>
<td>-164.25438</td>
</tr>
<tr>
<td>4</td>
<td>-164.25438</td>
</tr>
</tbody>
</table>

Generalized structural equation model

<table>
<thead>
<tr>
<th>Number of obs</th>
<th>74</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>mpg</td>
</tr>
<tr>
<td>Lower limit</td>
<td>17</td>
</tr>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
</tbody>
</table>

Log likelihood = -164.25438

| Coef.    | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|----------|-----------|------|------|---------------------|
| mpg      | wgt       |      |      |                     |
| -6.87305 | .700257   | -9.82| 0.000| -8.245529 -5.500572 |
| _cons    |           |      |      |                     |
| 41.49856 | 2.058384  | 20.16| 0.000| 37.4642  45.53291  |
| var(e.mpg)|         |      |      |                     |
| 14.78942 | 2.817609  | 10.18| 0.000| 10.18085 21.48414  |

Notes:

1. The reported coefficients and standard errors (SEs) match those reported by \texttt{tobit}.
2. \texttt{gsem} reports the point estimate of \(e.mpg\) as 14.78942. This is an estimate of \(\sigma^2\), the error variance. \texttt{tobit} reports an estimated \(\sigma\) as 3.845701. And \(\sqrt{14.78942} = 3.8457\).

**Fitting the model with the Builder**

Use the diagram in \textit{Fitting tobit regression models} above for reference.

1. Open the dataset and create the rescaled weight variable.
   
   In the Command window, type
   
   \begin{verbatim}
   . use https://www.stata-press.com/data/r16/auto
   . generate wgt = weight/1000
   \end{verbatim}

2. Open a new Builder diagram.
   
   Select menu item \texttt{Statistics > SEM (structural equation modeling) > Model building and estimation}.

3. Put the Builder in \texttt{gsem} mode by clicking on the \(\square\) button.

4. Create the independent variable.
   
   a. Select the Add observed variable tool, \(\square\).
   
   b. Click in the diagram about one-fourth of the way in from the left and half of the way up from the bottom.
   
   c. In the Contextual Toolbar, use the \texttt{Variable} control to select the variable \texttt{wgt}. 
5. Create the tobit response.
   a. Select the Add generalized response variable tool, □.
   b. Click about one-third of the way in from the right side of the diagram, to the right of the \texttt{wgt} rectangle.
   c. In the Contextual Toolbar, select Gaussian, Identity in the \textit{Family/Link} control (it may already be selected).
   d. In the Contextual Toolbar, use the \textit{Variable} control to select the variable \texttt{mpg}.
   e. In the Contextual Toolbar, click on the \textbf{Properties...} button.
   f. In the resulting \textit{Variable properties} dialog box, click on the \textbf{Censoring...} button in the \textit{Variable} tab.
   g. In the resulting \textit{Censoring} dialog box, select the \textit{Left censored} radio button. In the resulting \textit{Left censoring} box below, select the \textit{Constant} radio button (it may already be selected), and type 17 in the \textit{Constant} control.
   h. Click on \textbf{OK} in the \textit{Censoring} dialog box, and then click on \textbf{OK} in the \textit{Variable properties} dialog box. The Details pane will now show the censoring information for \texttt{mpg}.

6. Create a path from the independent variable to the dependent variable.
   a. Select the Add path tool, −.
   b. Click in the right side of the \texttt{wgt} rectangle (it will highlight when you hover over it), and drag a path to the left side of the \texttt{mpg} rectangle (it will highlight when you can release to connect the path).

7. Estimate.
   Click on the \textbf{Estimate} button, \textbullet, in the Standard Toolbar, and then click on \textbf{OK} in the resulting \textit{GSEM estimation options} dialog box.

   You can open a completed diagram in the Builder by typing
   \texttt{.webgetsem gsem_tobit}

\section*{Also see}

[SEM] Example 38g — Random-intercept and random-slope models (multilevel)
[SEM] Example 44g — Interval regression
[SEM] Example 45g — Heckman selection model
[SEM] Example 46g — Endogenous treatment-effects model
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
Interval regression is demonstrated using `intregxmpl.dta`:

```
. use https://www.stata-press.com/data/r16/intregxmpl
(Wages of women)
```

See *Structural models 1: Linear regression* in [SEM] *Intro 5* for background.

Remarks and examples

Remarks are presented under the following headings:

* Fitting interval regression models
* Fitting the model with the Builder

Fitting interval regression models

The first example in [R] `intreg` is

```
. intreg wage1 wage2 age c.age#c.age nev_mar rural school tenure
```

This model corresponds to

![Path diagram](image)

Interval measure information does not appear in the path diagram by default. It can be added to the path diagram by customizing the appearance of `wage1` in the Builder. The Builder reports the interval measure information for `wage1` in the Details pane.
To fit this model with `gsem`, we type

```
gsem wage1 <- age c.age#c.age nev_mar rural school tenure, > family(gaussian, udepvar(wage2))
```

Refining starting values:

```
Grid node 0:  log likelihood = -856.59446
```

Fitting full model:

```
Iteration 0:  log likelihood = -856.59446
Iteration 1:  log likelihood = -856.33321
Iteration 2:  log likelihood = -856.33293
Iteration 3:  log likelihood = -856.33293
```

```
Generalized structural equation model               Number of obs = 488
Lower response : wage1                            Uncensored = 0
Upper response : wage2                            Left-censored = 14
Family : Gaussian                                Right-censored = 6
Link : identity                                  Interval-cens. = 468
Log likelihood = -856.33293

|                 | Coef.  | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-----------------|--------|-----------|-------|-------|----------------------|
| wage1 age       | .7914438 | .4433604  | 1.79  | 0.074 | -.0775265 1.660414   |
| c.age#c.age     | -.0132624 | .0073028  | -1.82 | 0.069 | -.0275757 .0010509  |
| nev_mar         | -.2075022 | .8119581  | -0.26 | 0.798 | -1.798911 1.383906   |
| rural           | -3.043044 | .7757324  | -3.92 | 0.000 | -4.563452 -1.522637  |
| school          | 1.334721  | .1357873  | 9.83  | 0.000 | 1.068583 1.600859    |
| tenure          | .8000664  | .1045077  | 7.66  | 0.000 | .5952351 1.004898    |
| _cons           | -12.70238 | 6.367117  | -1.99 | 0.046 | -25.1817 -2230584   |
| var(e.wage1)    | 53.28454 | 3.693076  | 46.51635 | 61.0375 |
```

Notes:

1. Just like `intreg`, `gsem` requires two dependent variables for fitting interval regression models. The `udepvar()` suboption in `family(gaussian)` allows you to specify the dependent variable containing the upper-limit values for the interval regression. Consequently, the dependent variable participating in the path specification necessarily contains the lower-limit values.

2. Reported coefficients match those reported by `intreg`.

3. Reported standard errors (SEs) match those reported by `intreg`.

4. `gsem` reports the point estimate of `e.wage1` as 53.28454. This is an estimate of $\sigma^2$, the error variance. `intreg` reports an estimated $\sigma$ as 7.29962. And $\sqrt{53.28454} = 7.29962$.

**Fitting the model with the Builder**

Use the diagram in *Fitting interval regression models* above for reference.

1. Open the dataset.

   In the Command window, type

   ```
   use https://www.stata-press.com/data/r16/intregxmpl
   ```
2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in gsem mode by clicking on the $\mathbb{S}_m$ button.

4. Create the independent variables.
   Select the Add observed variables set tool, $\oplus$, and then click in the diagram about one-fourth of the way in from the left and one-fourth of the way up from the bottom.
   In the resulting dialog box,
   a. select the *Select variables* radio button (it may already be selected);
   b. type the following variable names in the *Variables* control: age, c.age#c.age, nev_mar, rural, school, and tenure;
   c. select *Vertical* in the *Orientation* control; and
   d. click on *OK*.
   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the interval responses.
   a. Select the Add generalized response variable tool, $\equiv$.
   b. Click about one-third of the way in from the right side of the diagram, to the right of the nev_mar rectangle.
   c. In the Contextual Toolbar, select Gaussian, Identity in the *Family/Link* control (it may already be selected).
   d. In the Contextual Toolbar, use the *Variable* control to select the variable wage1.
   e. In the Contextual Toolbar, click on the *Properties...* button.
   f. In the resulting *Variable properties* dialog box, click on the *Censoring...* button in the *Variable* tab.
   g. In the resulting *Censoring* dialog box, select the *Interval measured, depvar is lower boundary* radio button. In the resulting *Interval measured* box below, use the *Upper bound* control to select the variable wage2.
   h. Click on *OK* in the *Censoring* dialog box, and then click on *OK* in the *Variable properties* dialog box. The Details pane will now show that wage1 is the lower bound and wage2 is the upper bound of our interval measure.

6. Create paths from the independent variables to the dependent variable.
   a. Select the Add path tool, $\rightarrow$.
   b. Click in the right side of the age rectangle (it will highlight when you hover over it), and drag a path to the left side of the wage1 rectangle (it will highlight when you can release to connect the path).
c. Continuing with the tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable:

c.age#c.age -> wage1
dev_num -> wage1
rural -> wage1
school -> wage1
tenure -> wage1

7. Estimate.

Click on the Estimate button, in the Standard Toolbar, and then click on OK in the resulting GSEM estimation options dialog box.

You can open a completed diagram in the Builder by typing

`.webgetsem gsem_intreg`

Also see

[SEM] Example 38g — Random-intercept and random-slope models (multilevel)
[SEM] Example 43g — Tobit regression
[SEM] Example 45g — Heckman selection model
[SEM] Example 46g — Endogenous treatment-effects model
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
Description

To demonstrate selection models, we will use the following data:

```stata
use https://www.stata-press.com/data/r16/gsem_womenwk
(Fictional data on women and work)
 summarize
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>2000</td>
<td>36.208</td>
<td>8.28656</td>
<td>20</td>
<td>59</td>
</tr>
<tr>
<td>educ</td>
<td>2000</td>
<td>13.084</td>
<td>3.045912</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>married</td>
<td>2000</td>
<td>.6705</td>
<td>.4701492</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>children</td>
<td>2000</td>
<td>1.6445</td>
<td>1.398963</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>wage</td>
<td>1343</td>
<td>23.69217</td>
<td>6.305374</td>
<td>5.88497</td>
<td>45.80979</td>
</tr>
</tbody>
</table>

See Structural models 8: Dependencies between response variables and Structural models 9: Unobserved inputs, outputs, or both in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- The Heckman selection model as an SEM
- Fitting the Heckman selection model as an SEM
- Transforming results and obtaining rho
- Fitting the model with the Builder

The Heckman selection model as an SEM

We demonstrate below how `gsem` can be used to fit the Heckman selection model (Gronau 1974; Lewis 1974; Heckman 1976) and produce results comparable to those of Stata’s dedicated `heckman` command; see [R] heckman.

Our purpose is not to promote `gsem` as an alternative to `heckman`. We have two other purposes.

One is to show that `gsem` can be used to generalize the Heckman selection model to response functions other than linear and, in addition or separately, to include multilevel effects when such effects are present.

The other is to show how Heckman selection models can be included in more complicated SEMs.
For those unfamiliar with this model, it deals with a continuous outcome that is observed only when another equation determines that the observation is selected, and the errors of the two equations are allowed to be correlated. Subjects often choose to participate in an event or medical trial or even the labor market, and thus the outcome of interest might be correlated with the decision to participate. Heckman won a Nobel Prize for this work.

The model is sometimes cast in terms of female labor supply, but it obviously has broader application. Nevertheless, we will consider a female labor-supply example.

Women are offered employment at a wage of $w$,

$$w_i = X_i\beta + \epsilon_i$$

Not all women choose to work, and $w$ is observed only for those women who do work. Women choose to work if

$$Z_i\gamma + \xi_i > 0$$

where

$$\epsilon_i \sim N(0, \sigma^2)$$

$$\xi_i \sim N(0, 1)$$

$$\text{corr}(\epsilon, \xi) = \rho$$

More generally, we can think of this model as applying to any continuously measured outcome $w_i$, which is observed only if $Z_i\gamma + \xi_i > 0$. The important feature of the model is that the errors $\xi_i$ of the selection equation and the errors $\epsilon_i$ of the observed-data equation are allowed to be correlated.

The Heckman selection model can be recast as a two-equation SEM—one linear regression (for the continuous outcome) and the other censored regression (for selection)—and with a latent variable $L_i$ added to both equations. The latent variable is constrained to have variance 1 and to have coefficient 1 in the selection equation, leaving only the coefficient in the continuous-outcome equation to be estimated. For identification, the variance from the censored regression will be constrained to be equal to that of the linear regression. The results of doing this are the following:

1. Latent variable $L_i$ becomes the vehicle for carrying the correlation between the two equations.
2. All the parameters given above, namely, $\beta$, $\gamma$, $\sigma^2$, and $\rho$, can be recovered from the SEM estimates.
3. If we call the estimated parameters in the SEM formulation $\beta^*$, $\gamma^*$, and $\sigma^2*$, and let $\kappa$ denote the coefficient on $L_i$ in the continuous-outcome equation, then

$$\beta = \beta^*$$

$$\gamma = \gamma^*/\sqrt{\sigma^2* + 1}$$

$$\sigma^2 = \sigma^2* + \kappa^2$$

$$\rho = \kappa/\sqrt{(\sigma^2* + \kappa^2)(\sigma^2* + 1)}$$

This parameterization places no restriction on the range or sign of $\rho$. See Skrondal and Rabe-Hesketh (2004, 107–108).
Fitting the Heckman selection model as an SEM

We wish to fit the following Heckman selection model:

\[
\begin{array}{c}
\varepsilon_1 \\
\varepsilon_2 \\
\end{array}
\]

married
children
educ
age
selected
Gaussian
identity
ε1
\begin{array}{c}
\varepsilon_2 \\
\end{array}
wage
ε2
L
1
\]

What makes this a Heckman selection model is

1. the inclusion of latent variable \( L \) in both the continuous-outcome (wage) equation and the censored-outcome selection equation;
2. constraining the selected <- \( L \) path coefficient to be 1;
3. constraining the variance of \( L \) to be 1; and
4. constraining the error variances to be equal.

Before we can fit this model, we need to create new variables selected and notselected. selected will equal 0 if the woman works (wage is not missing) and missing otherwise. notselected is the complement of selected: it equals 0 if the woman does not work (wage is missing) and missing otherwise. selected and notselected will be used as the dependent variables in the censored regression, providing the equivalent of a scaled probit regression.

. generate selected = 0 if wage < .
(657 missing values generated)
. generate notselected = 0 if wage >= .
(1,343 missing values generated)
. tabulate selected notselected, missing

<table>
<thead>
<tr>
<th>selected</th>
<th>notselected</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1,343</td>
</tr>
<tr>
<td>.</td>
<td>657</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>657</td>
<td>1,343</td>
</tr>
</tbody>
</table>

Old-time Stata users may be worried that because wage is missing in so many observations, namely, all those corresponding to nonworking women, there must be something special we need to do so that \texttt{gsem} uses all the data. There is nothing special we need to do. \texttt{gsem} counts missing values on an equation-by-equation basis, so it will use all the data for the censored regression part of the model while simultaneously using only the working-woman subsample for the continuous-outcome (wage) part of the model. We use all the data for the censored regression because \texttt{gsem} understands the meaning of missing values in the censored dependent variables so long as one of them is nonmissing.
To fit this model in command syntax, we type

```
. gsem (wage <- educ age L)
>   (selected <- married children educ age L@1,
>       family(gaussian, udepvar(notselected)), var(L@1 e.wage@a e.selected@a)
```

Fitting fixed-effects model:

```
Fitting fixed-effects model:
Iteration 0: log likelihood = -5752.6506
Iteration 1: log likelihood = -5260.9961
Iteration 2: log likelihood = -5209.2571
Iteration 3: log likelihood = -5208.9039
Iteration 4: log likelihood = -5208.9038
Refining starting values:
Iterations 0: log likelihood = -5208.7006
Fitting full model:
Iteration 0: log likelihood = -5208.5322 (not concave)
Iteration 1: log likelihood = -5208.0269 (not concave)
Iteration 2: log likelihood = -5202.872 (not concave)
Iteration 3: log likelihood = -5202.0258
Iteration 4: log likelihood = -5201.6058
Iteration 5: log likelihood = -5201.5378
Iteration 6: log likelihood = -5201.5347
Iteration 7: log likelihood = -5201.5347
Iteration 8: log likelihood = -5201.5347
Generalized structural equation model Number of obs = 2,000
Response : wage Number of obs = 1,343
Family : Gaussian
Link : identity
Lower response : selected Number of obs = 2,000
Upper response : notselected Uncensored = 0
Family : Gaussian Left-censored = 657
Link : identity Right-censored = 1,343
Interval-cens. = 0
Log likelihood = -5178.3046
( 1) [selected]L = 1
( 2) - [/]var(e.wage) + [/]var(e.selected) = 0
( 3) [/]var(L) = 1

|               | Coef.   | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|---------------|---------|-----------|-------|-------|---------------------|
| wage          |         |           |       |       |                     |
| educ          | .9899512| .0532552  | 18.59 | 0.000 | .8855729            |
| age           | .2131282| .020602   | 10.35 | 0.000 | .172749             |
| L             | 5.923736| .1846818  | 32.08 | 0.000 | 5.561767            |
| _cons         | .4859114| 1.076865  | 0.45  | 0.652 | -1.624705           |
| selected      |         |           |       |       |                     |
| married       | .6242746| .1054319  | 5.92  | 0.000 | .4176319            |
| children      | .6152095| .0652002  | 9.44  | 0.000 | .4874196            |
| educ          | .0781542| .0162868  | 4.80  | 0.000 | .0462327            |
| age           | .0511983| .0065637  | 7.71  | 0.000 | .0381901            |
| L             | -3.493217| .3730379  | -9.36 | 0.000 | -4.224357           |
| _cons         | -3.493217| .3730379  | -9.36 | 0.000 | -4.224357           |
| var(L)        | 1       |           |       |       |                     |
| var(e.wage)   | .9664635| .2689653  |       |       | .560141             |
| var(e.selected) | .9664635| .2689653  |       |       | .560141             |
Notes:

1. Some of the estimated coefficients and parameters above will match those reported by the `heckman` command and others will not. The above parameters are in the transformed structural equation modeling metric. That metric can be transformed back to the Heckman metric and results will match. The relationship to the Heckman metric is

\[
\begin{align*}
\beta &= \beta^* \\
\gamma &= \gamma^* / \sqrt{\sigma^2 + 1} \\
\sigma^2 &= \sigma^2 + \kappa^2 \\
\rho &= \kappa / \sqrt{(\sigma^2 + \kappa^2)(\sigma^2 + 1)}
\end{align*}
\]

2. \(\beta\) refers to the coefficients on the continuous-outcome (wage) equation. We can read those coefficients directly, without transformation except that we ignore the `wage <- L` path:

\[
\text{wage} = 0.9900 \text{ educ} + 0.2131 \text{ age} + 0.4859
\]

3. \(\gamma\) refers to the selection equation, and because \(\gamma = \gamma^* / \sqrt{\sigma^2 + 1}\), we must divide the reported coefficients by the square root of \(\sigma^2 + 1\). What has happened here is that the scaled probit has variance \(\sigma^2 + 1\), and we are merely transforming back to the standard probit model, which has variance 1. The results are

\[
\Pr(\text{selected} = 0) = \Phi(0.4452 \text{ married} + 0.4387 \text{ children} + 0.0557 \text{ educ} + 0.0365 \text{ age} - 2.4910)
\]

4. To calculate \(\rho\), we first calculate \(\sigma^2 = \sigma^2 + \kappa^2\) and then calculate \(\rho = \kappa / \sqrt{\sigma^2 (\sigma^2 + 1)}\):

\[
\begin{align*}
\sigma^2 &= 0.9664 + 5.9237^2 = 36.0571 \\
\rho &= 5.9237 / \sqrt{36.0571} = 0.7035
\end{align*}
\]

5. These transformed results match the results that would have been reported had we typed

\[
. \text{heckman wage educ age, select(married children educ age)}
\]

(output omitted)

6. There is an easier way to obtain the transformed results than by hand, and the easier way provides standard errors. That is the subject of the next section.

### Transforming results and obtaining rho

We can use Stata’s `nlcom` command to perform the transformations we made by hand above, and we can obtain standard errors.

Let’s start by obtaining \(\sigma^2\) and \(\rho\). To remind you, the formulas are

\[
\begin{align*}
\sigma^2 &= \sigma^2 + \kappa^2 \\
\rho &= \kappa / \sqrt{\sigma^2 (\sigma^2 + 1)}
\end{align*}
\]

We must describe these two formulas in a way that `nlcom` can understand. The Stata notation for parameters \(\sigma^2\) and \(\kappa\) fit by `gsem` is

\[
\begin{align*}
\sigma^2: \_b[\text{var(e.wage)}] \\
\kappa: \_b[\text{wage:L}]
\end{align*}
\]
We cannot remember that notation; however, we can type `gsem, coeflegend` to be reminded. We now have all that we need to obtain the estimates of $\sigma^2$ and $\rho$. Because `heckman` reports $\sigma$ rather than $\sigma^2$, we will tell `nlcom` to report the $\sqrt{\sigma^2}$:

\[
\begin{align*}
\text{nlcom (sigma: sqrt(_b[/var(e.wage)] +_b[wage:L]^2))} \\
> & (rho: _b[wage:L]/(sqrt((_b[/var(e.wage)]+1)*(_b[/var(e.wage)] + _b[wage:L]^2)))) \\
> & \sigma: sqrt(_b[/var(e.wage)] +_b[wage:L]^2) \\
> & \rho: _b[wage:L]/(sqrt((_b[/var(e.wage)]+1)*(_b[/var(e.wage)] + _b[wage:L]^2))) \\
\end{align*}
\]

|                  | Coef.       | Std. Err. |    z  | P>|z| | [95% Conf. Interval] |
|------------------|-------------|-----------|-------|------|----------------------|
| sigma            | 6.004758    | 0.1656471 | 36.25 | 0.000| 5.680095, 6.32942    |
| rho              | 0.703489    | 0.0511861 | 13.74 | 0.000| 0.603166, 0.8038119  |

The output above nearly matches what `heckman` reports. `heckman` does not report the test statistics and $p$-values for these two parameters. In addition, the confidence interval that `heckman` reports for $\rho$ will differ slightly from the above and is better. `heckman` uses a method that will not allow $\rho$ to be outside of $-1$ and 1, whereas `nlcom` is simply producing a confidence interval for the calculation we requested and in absence of the knowledge that the calculation corresponds to a correlation coefficient. The same applies to the confidence interval for $\sigma$, where the bounds are 0 and infinity.

To obtain the coefficients and standard errors for the selection equation, we type

\[
\begin{align*}
\text{nlcom (married: _b[selected:married]/sqrt(_b[/var(e.wage)]+1))} \\
& (children: _b[selected:children]/sqrt(_b[/var(e.wage)]+1)) \\
& (educ: _b[selected:educ]/sqrt(_b[/var(e.wage)]+1)) \\
& (age: _b[selected:age]/sqrt(_b[/var(e.wage)]+1)) \\
\end{align*}
\]

|                  | Coef.       | Std. Err. |    z  | P>|z| | [95% Conf. Interval] |
|------------------|-------------|-----------|-------|------|----------------------|
| married          | 0.445177    | 0.0673953 | 6.61  | 0.000| 0.3130847, 0.5772693 |
| children         | 0.4387126   | 0.0277788 | 15.79 | 0.000| 0.3842671, 0.4931581 |
| educ             | 0.0557326   | 0.0107348 | 5.19  | 0.000| 0.0346927, 0.0767725 |
| age              | 0.0365101   | 0.0041534 | 8.79  | 0.000| 0.0283696, 0.0446505 |

The above output matches what `heckman` reports.

**Fitting the model with the Builder**

Use the diagram in *Fitting the Heckman selection model as an SEM* above for reference.

1. Open the dataset and create the selection variable.

   In the Command window, type

   ```stata
   use https://www.stata-press.com/data/r16/gsem_womenwk
   generate selected = 0 if wage < .
   generate notselected = 0 if wage >= .
   ```

2. Open a new Builder diagram.

   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation.**
3. Put the Builder in *gsem* mode by clicking on the \( \mathcal{G} \) button.

4. Create the independent variables.

Select the Add observed variables set tool, \( \mathcal{O} \), and then click in the diagram about one-fourth of the way in from the left and one-fourth of the way up from the bottom.

In the resulting dialog box,

a. select the *Select variables* radio button (it may already be selected);

b. use the *Variables* control to select the variables *married*, *children*, *educ*, and *age* in this order;

c. select *Vertical* in the *Orientation* control;

d. click on **OK**.

If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the generalized response for selection.

a. Select the Add generalized response variable tool, \( \mathcal{A} \).

b. Click about one-third of the way in from the right side of the diagram, to the right of the *married* rectangle.

c. In the Contextual Toolbar, select *Gaussian*, *Identity* in the *Family/Link* control (it may already be selected).

d. In the Contextual Toolbar, select *selected* in the *Variable* control.

e. In the Contextual Toolbar, click on the **Properties...** button.

f. In the resulting *Variable properties* dialog box, click on the **Censoring...** button in the *Variable* tab.

g. In the resulting *Censoring* dialog box, select the *Interval measured, depvar is lower boundary* radio button. In the resulting *Interval measured* box below, use the *Upper bound* control to select the variable *notselected*.

h. Click on **OK** in the *Censoring* dialog box, and then click on **OK** in the *Variable properties* dialog box. The Details pane will now show *selected* as the lower bound and *notselected* as the upper bound of our interval measure.

6. Create the endogenous *wage* variable.

a. Select the Add observed variable tool, \( \mathcal{O} \), and then click about one-third of the way in from the right side of the diagram, to the right of the *age* rectangle.

b. In the Contextual Toolbar, select *wage* with the *Variable* control.

7. Create paths from the independent variables to the dependent variables.

a. Select the Add path tool, \( \rightarrow \).

b. Click in the right side of the *married* rectangle (it will highlight when you hover over it), and drag a path to the left side of the *selected* rectangle (it will highlight when you can release to connect the path).

c. Continuing with the \( \rightarrow \) tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable:
8. Clean up the direction of the error terms.

We want the error for \textit{selected} to be above the rectangle and the error for \textit{wage} to be below the rectangle, but it is likely they have been created in other directions.

   a. Choose the Select tool, \textbf{Select tool}.
   b. Click in the \textit{selected} rectangle.
   c. Click on one of the \textbf{Error rotation} buttons, \textbf{Error rotation}, in the Contextual Toolbar until the error is above the rectangle.
   d. Click in the \textit{wage} rectangle.
   e. Click on one of the \textbf{Error rotation} buttons, \textbf{Error rotation}, in the Contextual Toolbar until the error is below the rectangle.

9. Create the latent variable.

   a. Select the Add latent variable tool, \textbf{Add latent variable tool}, and then click at the far right of the diagram and vertically centered between the \textit{selected} and \textit{wage} variables.
   b. In the Contextual Toolbar, type \textit{L} in the \textit{Name} control and press \textit{Enter}.

10. Draw paths from the latent variable to each endogenous variable.

   a. Select the Add path tool, \textbf{Add path tool}.
   b. Click in the upper left quadrant of the \textit{L} oval, and drag a path to the right side of the \textit{selected} rectangle.
   c. Continuing with the \textbf{Add path tool}, create another path by clicking first in the lower-left quadrant of the \textit{L} oval and dragging a path to the right side of the \textit{wage} rectangle.

11. Place constraints on the variances and on the path from L to \textit{selected}.

   a. Choose the Select tool, \textbf{Select tool}.
   b. Click on the \textit{L} oval. In the Contextual Toolbar, type 1 in the \textbf{Box} box and press \textit{Enter}.
   c. Click on the error oval attached to the \textit{wage} rectangle. In the Contextual Toolbar, type a in the \textbf{Box} box and press \textit{Enter}.
   d. Click on the error oval attached to the \textit{selected} rectangle. In the Contextual Toolbar, type a in the \textbf{Box} box and press \textit{Enter}.
   e. Click on the path from \textit{L} to \textit{selected}. In the Contextual Toolbar, type 1 in the \textbf{Box} box and press \textit{Enter}.

12. Clean up the location of the paths.

If you do not like where a path has been connected to its variables, use the Select tool, \textbf{Select tool}, to click on the path, and then simply click on where it connects to a rectangle and drag the endpoint.

Click on the **Estimate** button, ![Estimate button](image), in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

You can open a completed diagram in the Builder by typing

```
.webgetsem gsem_select
```

References


Heckman, J. 1976. The common structure of statistical models of truncation, sample selection and limited dependent variables and a simple estimator for such models. *Annals of Economic and Social Measurement* 5: 475–492.


Also see

[SEM] **Example 34g** — Combined models (generalized responses)

[SEM] **Example 46g** — Endogenous treatment-effects model

[SEM] **Intro 5** — Tour of models

[SEM] **gsem** — Generalized structural equation model estimation command
**Example 46g — Endogenous treatment-effects model**

**Description**

To illustrate the treatment-effects model, we use the following data:

```stata
use https://www.stata-press.com/data/r16/gsem_union3
(NLSY 1972)
describe
```

Contains data from https://www.stata-press.com/data/r16/gsem_union3.dta

<table>
<thead>
<tr>
<th>obs:</th>
<th>1,693</th>
</tr>
</thead>
<tbody>
<tr>
<td>vars:</td>
<td>24</td>
</tr>
<tr>
<td>29 Mar 2018 11:30</td>
<td></td>
</tr>
<tr>
<td>(_dta has notes)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>idcode</td>
<td>int</td>
<td>%8.0g</td>
<td></td>
<td>NLS ID</td>
</tr>
<tr>
<td>year</td>
<td>int</td>
<td>%8.0g</td>
<td></td>
<td>interview year</td>
</tr>
<tr>
<td>birth_yr</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>birth year</td>
</tr>
<tr>
<td>age</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>age in current year</td>
</tr>
<tr>
<td>race</td>
<td>byte</td>
<td>%8.0g</td>
<td>racelbl</td>
<td>race</td>
</tr>
<tr>
<td>msp</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if married, spouse present</td>
</tr>
<tr>
<td>nev_mar</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if never married</td>
</tr>
<tr>
<td>grade</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>current grade completed</td>
</tr>
<tr>
<td>collgrad</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if college graduate</td>
</tr>
<tr>
<td>not_smsa</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if not SMSA</td>
</tr>
<tr>
<td>c_city</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if central city</td>
</tr>
<tr>
<td>south</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if south</td>
</tr>
<tr>
<td>ind_code</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>industry of employment</td>
</tr>
<tr>
<td>occ_code</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>occupation</td>
</tr>
<tr>
<td>union</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if union</td>
</tr>
<tr>
<td>wks_ue</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>weeks unemployed last year</td>
</tr>
<tr>
<td>ttl_exp</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>total work experience</td>
</tr>
<tr>
<td>tenure</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>job tenure, in years</td>
</tr>
<tr>
<td>hours</td>
<td>int</td>
<td>%8.0g</td>
<td></td>
<td>usual hours worked</td>
</tr>
<tr>
<td>wks_work</td>
<td>int</td>
<td>%8.0g</td>
<td></td>
<td>weeks worked last year</td>
</tr>
<tr>
<td>ln_wage</td>
<td>float</td>
<td>%9.0g</td>
<td>ln(wage/GNP deflator)</td>
<td></td>
</tr>
<tr>
<td>wage</td>
<td>double</td>
<td>%10.0g</td>
<td></td>
<td>real wage</td>
</tr>
<tr>
<td>black</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>race black</td>
</tr>
<tr>
<td>smsa</td>
<td>byte</td>
<td>%8.0g</td>
<td></td>
<td>1 if SMSA</td>
</tr>
</tbody>
</table>

Sorted by: idcode
des
_dta:

2. These data from 1972 were subsetted for purposes of demonstration.

See *Structural models 8: Dependencies between response variables* and *Structural models 9: Unobserved inputs, outputs, or both* in [SEM] *Intro 5* for background.
Remarks and examples

Remarks are presented under the following headings:

Fitting the treatment-effects model
Fitting the model with the Builder

Fitting the treatment-effects model

We wish to fit the following model:

\[
\begin{align*}
1.\text{south} & \quad \varepsilon_1 \quad a \\
1.\text{black} & \quad \text{Gaussian} \\
tenure & \quad \text{identity} \\
age & \\
grade & \\
1.\text{smsa} & \\
1.\text{union} & \\
\end{align*}
\]

We wish to estimate the "treatment effect" of being a union member. That is, we speculate that union membership has an effect on wages, and we want to measure that effect. The problem would be easy if we had data on the same workers from two different but nearly identical universes, one in which the workers were not union members and another in which they were.

The model above is similar to the Heckman selection model we fit in [SEM] Example 45g. The differences are that the continuous variable (wage) is observed in all cases and that we have a path from the treatment indicator (previously selection, now treatment) to the continuous variable. Just as with the Heckman selection model, we allow for correlation by introducing a latent variable with model identification constraints.

Before we can fit this model, we need to create new variables llunion and ulunion. llunion will equal 0 if union is 1 and missing otherwise. ulunion is the complement of llunion: it equals 0 if union is 0 and missing otherwise. llunion and ulunion will be used as the dependent variables in the treatment equation, providing the equivalent of a scaled probit regression.

\[
\begin{align*}
\text{generate llunion} &= 0 \text{ if } \text{union} = 1 \\
(1,433 \text{ missing values generated}) \\
\text{generate ulunion} &= 0 \text{ if } \text{union} = 0 \\
(709 \text{ missing values generated})
\end{align*}
\]
We can now fit this model with command syntax by typing

```
  gsem (wage <- age grade i.smsa i.black tenure 1.union L)
  >  (llunion <- i.black tenure i.south L@1,
  >     family(gaussian, udepvar(ulunion))),
  >     var(L@1 e.wage@a e.llunion@a)
```

Fitting fixed-effects model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-3376.1731</td>
</tr>
<tr>
<td>1</td>
<td>-3096.7893</td>
</tr>
<tr>
<td>2</td>
<td>-3061.7875</td>
</tr>
<tr>
<td>3</td>
<td>-3061.497</td>
</tr>
<tr>
<td>4</td>
<td>-3061.497</td>
</tr>
</tbody>
</table>

Refining starting values:

<table>
<thead>
<tr>
<th>Grid node</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-3064.9724</td>
</tr>
</tbody>
</table>

Fitting full model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-3064.9427</td>
</tr>
<tr>
<td>1</td>
<td>-3060.3304</td>
</tr>
<tr>
<td>2</td>
<td>-3055.0099</td>
</tr>
<tr>
<td>3</td>
<td>-3096.6772</td>
</tr>
<tr>
<td>4</td>
<td>-3062.6735</td>
</tr>
<tr>
<td>5</td>
<td>-3054.2853</td>
</tr>
<tr>
<td>6</td>
<td>-3051.673</td>
</tr>
<tr>
<td>7</td>
<td>-3051.575</td>
</tr>
<tr>
<td>8</td>
<td>-3051.575</td>
</tr>
<tr>
<td>9</td>
<td>-3051.575</td>
</tr>
</tbody>
</table>

Generalized structural equation model

<table>
<thead>
<tr>
<th>Response</th>
<th>wage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
</tbody>
</table>

Censoring of obs:

- Lower response: llunion
  - Uncensored = 0
  - Left-censored = 957
- Upper response: ulunion
  - Right-censored = 253
- Link
  - Interval-censored = 0

Log likelihood = -3051.575

| ( 1) [llunion]L = 1
| ( 2) - [1]var(e.wage) + [1]var(e.llunion) = 0
| ( 3) [1]var(L) = 1

| Coef.  | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------|-----------|-------|------|---------------------|
| wage   |           |       |      |                     |
| age    | 0.1487409 | 0.0193291 | 7.70 | 0.000 | 0.1108566 | 0.1866252 |
| grade  | 0.4205658 | 0.0293577 | 14.33 | 0.000 | 0.3630258 | 0.4781057 |
| 1.smsa | 0.9117044 | 0.1249041 | 7.30 | 0.000 | 0.6668969 | 1.156512  |
| 1.black| -0.7882472 | 0.1367077 | 5.77 | 0.000 | -1.056189 | -0.520305 |
| tenure | 0.1524015 | 0.0369595 | 4.12 | 0.000 | 0.0799621 | 0.2248408 |
| 1.union| 2.945816  | 0.2749549 | 10.71 | 0.000 | 2.406914  | 3.484718  |
| L      | -1.706795 | 0.1288024 | -13.25 | 0.000 | -1.959243 | -1.454347 |
| _cons  | -4.351572 | 0.5283952 | -8.24 | 0.000 | -5.387207 | -3.315936 |

| llunion|          |       |      |                     |
| 1.black| 0.6704049 | 0.148057 | 4.53 | 0.000 | 0.3802185 | 0.9605913 |
| tenure | 0.1282024 | 0.0357986 | 3.58 | 0.000 | 0.0580384 | 0.1983664 |
| 1.south| -0.8542673 | 0.136439 | -6.26 | 0.000 | -1.121683 | -0.5868518 |
| L      | 1 (constrained) |          |      |                     |
| _cons  | -1.302676 | 0.1407538 | -9.25 | 0.000 | -1.578548 | -1.026804 |
Example 46g — Endogenous treatment-effects model

<table>
<thead>
<tr>
<th>var(L)</th>
<th>1 (constrained)</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(e.wage)</td>
<td>1.163821 .2433321 .7725324 1.753298</td>
</tr>
<tr>
<td>var(e.llun-n)</td>
<td>1.163821 .2433321 .7725324 1.753298</td>
</tr>
</tbody>
</table>

Notes:

1. The treatment effect is measured by the coefficient on the path treatment_variable -> continuous_variable or, in our case, 1.union->wage. It is estimated to be 2.9458, which is practically large and statistically significant.

2. The interpretation formulas are the same as for the Heckman selection model in [SEM] Example 45g, namely,

\[
\beta = \beta^* \\
\gamma = \gamma^*/\sqrt{\sigma^2 + 1} \\
\sigma^2 = \sigma^2* + \kappa^2 \\
\rho = \kappa/\sqrt{(\sigma^2 + \kappa^2)(\sigma^2 + 1)}
\]

To remind you, \(\beta\) are the coefficients in the continuous-outcome equation, \(\gamma\) are the coefficients in the treatment equation, \(\sigma^2\) is the variance of the error in the continuous-outcome equation, and \(\rho\) is the correlation between the errors in the treatment and continuous-outcome equations.

3. In the output above, \(\sigma^2* (\text{var(e.wage)})\) is 1.1638 and \(\kappa\) (the path coefficient on wage<-L) is -1.7068. In [SEM] Example 45g, we calculated \(\rho\) by hand and then showed how to use the software to obtain the value and its standard error. This time, we will go right to the software.

After obtaining symbolic names by typing `gsem, coeflegend`, we type the following to obtain \(\rho\):

```
    . nlcom (rho: _b[wage:L]/(sqrt(_b[/var(e.wage)] + 1)*sqrt(_b[/var(e.wage)] + _b[wage:L]^2)))
        rho: _b[wage:L]/(sqrt(_b[/var(e.wage)] + 1)*sqrt(_b[/var(e.wage)] + _b[wage:L]^2))
```

| Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|----------------|---------------|------------|-----------------|
| rho            | -0.574648     | 0.060969   | -9.43 0.000     | -0.6941451 -0.4551509 |

4. We can obtain the untransformed treatment coefficients just as we did in [SEM] Example 45g.

5. Just as with the Heckman selection model, with `gsem`, the treatment-effects model can be applied to generalized outcomes and include multilevel effects. See Skrondal and Rabe-Hesketh (2004, chap. 14.5) for an example with a Poisson response function.

### Fitting the model with the Builder

Use the diagram in *Fitting the treatment-effects model* above for reference.

1. Open the dataset.

In the Command window, type

```
    . use https://www.stata-press.com/data/r16/gsem_union3
    . generate llunion = 0 if union == 1
    . generate ulunion = 0 if union == 0
```
2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in gsem mode by clicking on the \( \text{gm} \) button.

4. Create the independent variables.
   Select the Add observed variables set tool, \( \varnothing \), and then click in the diagram about one-fourth of the way in from the left and one-fourth of the way up from the bottom.

   In the resulting dialog box,
   a. select the Select variables radio button (it may already be selected);
   b. type 1.south, 1.black, tenure, age, grade, 1.smsa, and 1.union in the Variables control (typing 1.varname rather than using the \( \Downarrow \) button to create i.varname prevents the rectangle corresponding to the base category for these binary variables from being created);
   c. select Vertical in the Orientation control; and
   d. click on OK.

   If you wish, move the set of variables by clicking on any variable and dragging it.

5. Create the generalized response for l1union.
   a. Select the Add generalized response variable tool, \( \varnothing \).
   b. Click about one-third of the way in from the right side of the diagram, to the right of the 1.black rectangle.
   c. In the Contextual Toolbar, select Gaussian, Identity in the Family/Link control (it may already be selected).
   d. In the Contextual Toolbar, select l1union in the Variable control.
   e. In the Contextual Toolbar, click on the Properties... button.
   f. In the resulting Variable properties dialog box, click on the Censoring... button in the Variable tab.
   g. In the resulting Censoring dialog box, select the Interval measured, depvar is lower boundary radio button. In the resulting Interval measured box below, use the Upper bound control to select the variable ulunion.
   h. Click on OK in the Censoring dialog box, and then click on OK in the Variable properties dialog box. The Details pane will now show l1union as the lower bound and ulunion as the upper bound for our interval measure.

6. Create the endogenous wage variable.
   a. Select the Add observed variable tool, \( \varnothing \), and then click about one-third of the way in from the right side of the diagram, to the right of the grade rectangle.
   b. In the Contextual Toolbar, select wage with the Variable control.

7. Create paths from the independent variables to the dependent variables.
   a. Select the Add path tool, \( \sim \).
   b. Click in the right side of the 1.south rectangle (it will highlight when you hover over it), and drag a path to the left side of the l1union rectangle (it will highlight when you can release to connect the path).
c. Continuing with the tool, create the following paths by clicking first in the right side of the rectangle for the independent variable and dragging it to the left side of the rectangle for the dependent variable:

- 1.black $\rightarrow$ llunion
- tenure $\rightarrow$ llunion
- 1.black $\rightarrow$ wage
- tenure $\rightarrow$ wage
- age $\rightarrow$ wage
- grade $\rightarrow$ wage
- 1.smsa $\rightarrow$ wage
- 1.union $\rightarrow$ wage

8. Clean up the direction of the error terms.

We want the error for llunion to be above the rectangle and the error for wage to be below the rectangle, but it is likely they have been created in other directions.

a. Choose the Select tool.

b. Click in the llunion rectangle.

c. Click on one of the Error rotation buttons, in the Contextual Toolbar until the error is above the rectangle.

d. Click in the wage rectangle.

e. Click on one of the Error rotation buttons, , in the Contextual Toolbar until the error is below the rectangle.

9. Create the latent variable.

a. Select the Add latent variable tool, and then click at the far right of the diagram and vertically centered between the llunion and wage variables.

b. In the Contextual Toolbar, type L in the Name control and press Enter.

10. Draw paths from the latent variable to each endogenous variable.

a. Select the Add path tool.

b. Click in the upper-left quadrant of the L oval, and drag a path to the right side of the llunion rectangle.

c. Continuing with the tool, create another path by clicking first in the lower-left quadrant of the L oval and dragging a path to the right side of the wage rectangle.

11. Place constraints on the variances and on the path from L to llunion.

a. Choose the Select tool.

b. Click on the L oval. In the Contextual Toolbar, type 1 in the box and press Enter.

c. Click on the error oval attached to the wage rectangle. In the Contextual Toolbar, type a in the box and press Enter.

d. Click on the error oval attached to the llunion rectangle. In the Contextual Toolbar, type a in the box and press Enter.

e. Click on the path from L to llunion. In the Contextual Toolbar, type 1 in the box and press Enter.
12. Clean up the location of the paths.

If you do not like where a path has been connected to its variables, use the Select tool, to click on the path, and then simply click on where it connects to a variable and drag the endpoint.


Click on the **Estimate** button, in the Standard Toolbar, and then click on **OK** in the resulting **GSEM estimation options** dialog box.

You can open a completed diagram in the Builder by typing

```
.webgetsem gsem_treat
```

---

**References**


---

**Also see**

- **[SEM] Example 34g** — Combined models (generalized responses)
- **[SEM] Example 45g** — Heckman selection model
- **[SEM] Intro 5** — Tour of models
- **[SEM] gsem** — Generalized structural equation model estimation command
Description

In this example, we demonstrate how to fit parametric survival models with \texttt{gsem}. Specifically, in this example, we fit an exponential model, but much of the discussion applies to Weibull, gamma, loglogistic, and lognormal models as well.

\begin{verbatim}
. use https://www.stata-press.com/data/r16/gsem_kva
(Generator experiment)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_kva.dta
obs: 12 Generator experiment
vars: 3 23 Jan 2019 21:41
(_dta has notes)

variable name type format label variable label
 failtime int %9.0g Time until failure (hrs.)
 load byte %9.0g Overload (kVA)
 bearings byte %9.0g Has new bearings

Sorted by: 
. notes
_dta:
1. Artificial experimental data on two types of bearings in emergency generators.
2. The purpose is to compare the ability to withstand overloads of new-style bearings to old-style bearings.
3. All generators are run until failure.
4. Experiments were performed under overloads of 20, 25, 30, 35, and 40 kVA.
\end{verbatim}

See \textit{Structural models 7: Survival models} in [SEM] \textbf{Intro 5} for background.

Remarks and examples

Remarks are presented under the following headings:

\begin{itemize}
\item \textit{Fitting the exponential model}
\item \textit{Obtaining hazard ratios}
\item \textit{Fitting the model with the Builder}
\end{itemize}
Fitting the exponential model

We wish to fit the following model:

\[
\text{failtime} \sim \text{exponential} \left[ \log(\text{load}) + \beta \right],
\]

That is, we wish to fit an exponential model in which the failure time of bearings (\text{failtime}) depends on the amount of overload (\text{load}) and whether the bearings are the new style (\text{bearings} = 1) or the old style (\text{bearings} = 0).

To fit this model, we use the \texttt{gsem} command with the \texttt{exponential} option.

\[
\texttt{. gsem (failtime <- load i.bearings), exponential}
\]

\[
\begin{array}{c}
\text{Iteration 0: log likelihood = -84108.99} \\
\text{Iteration 1: log likelihood = -67.363768} \\
\text{Iteration 2: log likelihood = -66.855013} \\
\text{Iteration 3: log likelihood = -62.300033} \\
\text{Iteration 4: log likelihood = -62.227842} \\
\text{Iteration 5: log likelihood = -62.227568} \\
\text{Iteration 6: log likelihood = -62.227568}
\end{array}
\]

Generalized structural equation model Number of obs = 12
Response : failtime No. of failures = 12
Family : exponential Time at risk = 896
Form : proportional hazards
Link : log
Log likelihood = -62.227568

|       | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------|--------|-----------|-------|-------|---------------------|
| failtime |       |           |       |       |                     |
| load   | 0.0611166 | 0.0354318 | 1.72  | 0.085 | [.0083284, .1305616] |
| 1.bearings | -0.2194825 | 0.5773503 | -0.38 | 0.704 | [-1.351068, .9121033] |
| _cons  | -5.756595  | 1.056403  | -5.45 | 0.000 | [-7.827106, -3.686084] |

Notes:

1. By default, exponential models are fit in the proportional-hazards metric. The \texttt{exponential} option can be replaced with \texttt{family(exponential, aft)} if you want to fit the model in the accelerated failure-time metric.

2. \texttt{gsem} reports coefficients. These coefficients match those reported by the equivalent \texttt{streg} command,

\[
\texttt{. stset failtime} \\
\texttt{. streg load bearings, distribution(exponential) nohr}
\]

3. Without the \texttt{nohr} option, \texttt{streg} reports hazard ratios, which are the exponentiated coefficients.
Obtaining hazard ratios

The `estat eform` command reports exponentiated coefficients for models fit with `gsem`, so we can obtain hazard ratios as follows.

```
. estat eform
```

|          | exp(b)  | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|----------|---------|-----------|------|------|---------------------|
| `failtime` |         |           |      |      |                     |
| `load`   | 1.063023| .0376648  | 1.72 | 0.085| .9917062 1.139468   |
| `1.bearings` | .8029342| .4635743  | -0.38| 0.704| .2589635 2.489553   |
| `_cons`  | .0031619| .0033402  | -5.45| 0.000| .0003988 .02507     |

The hazard ratio of 0.80 for `bearings = 1` indicates that the predicted hazard of failure for the new style of bearing is 80% of the hazard for a bearing of the old type, provided that they have the same loading.

Fitting the model with the Builder

Use the diagram in *Fitting the exponential model* above for reference.

1. Open the dataset.
   In the Command window, type
   ```
   . use https://www.stata-press.com/data/r16/gsem_kva
   ```
2. Open a new Builder diagram.
   Select menu item Statistics > SEM (structural equation modeling) > Model building and estimation.
3. Put the Builder in `gsem` mode by clicking on the \( \mathbb{G} \) button.
4. Create the exponential regression component for `failtime`.
   Select the Add regression component tool, \( \mathbb{B} \), and then click in the diagram about one-third of the way in from the left and halfway down.
   In the resulting dialog box,
   a. select `failtime` in the Dependent variable control;
   b. check Make response generalized;
   c. select Exponential, Log in the Family/Link control;
   d. select the Select variables radio button (it may already be selected);
   e. use the Independent variables control to select the variable `load`;
   f. include the levels of the factor variable `bearings` by clicking on the \( \mathbb{G} \) button next to the Independent variables control. In the resulting dialog box, select the Factor variable radio button, select Main effect in the Specification control, and select `bearings` in the Variables control for Variable 1. Click on Add to varlist, and then click on OK;
   g. select Left in the Independent variables’ direction control;
   h. click on OK.
   If you wish, move the component by clicking on any variable and dragging it.
5. Clean up.
   The box for $\text{failtime}$ is created closer to the independent variables than it is in the example diagram. Use the Select tool, $\text{Select Tool}$, and click on the box for $\text{failtime}$. Drag it to the right to allow more space for results along the paths.

   Click on the $\text{Estimate}$ button, $\text{Estimate}$, in the Standard Toolbar, and then click on $\text{OK}$ in the resulting $\text{GSEM estimation options}$ dialog box.

   You can open a completed diagram in the Builder by typing
   
   $\text{. webgetsem gsem\_exp}$

**Also see**

[SEM] Example 48g — Loglogistic survival model with censored and truncated data

[SEM] Example 49g — Multiple-group Weibull survival model

[SEM] Intro 5 — Tour of models

[SEM] gsem — Generalized structural equation model estimation command

[SEM] estat eform — Display exponentiated coefficients
Example 48g — Loglogistic survival model with censored and truncated data

Description

In this example, we demonstrate how to fit a survival model to data that are both left-truncated and right-censored.

. use https://www.stata-press.com/data/r16/gsem_diet
   (Diet data with dates)
. describe

Contains data from https://www.stata-press.com/data/r16/gsem_diet.dta
   obs: 337 Diet data with dates
   vars: 11 16 Jan 2019 11:24
   (_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>type</th>
<th>format</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>%9.0g</td>
<td>Subject identity number</td>
</tr>
<tr>
<td>fail</td>
<td>byte</td>
<td>%8.0g</td>
<td>Outcome (CHD = 1 3 13)</td>
</tr>
<tr>
<td>job</td>
<td>byte</td>
<td>%8.0g</td>
<td>Occupation</td>
</tr>
<tr>
<td>month</td>
<td>byte</td>
<td>%8.0g</td>
<td>month of survey</td>
</tr>
<tr>
<td>energy</td>
<td>float</td>
<td>%9.0g</td>
<td>Total energy (1000kcal/day)</td>
</tr>
<tr>
<td>height</td>
<td>float</td>
<td>%9.0g</td>
<td>Height (cm)</td>
</tr>
<tr>
<td>weight</td>
<td>float</td>
<td>%9.0g</td>
<td>Weight (kg)</td>
</tr>
<tr>
<td>hienergy</td>
<td>byte</td>
<td>%9.0g</td>
<td>Indicator for high energy</td>
</tr>
<tr>
<td>doe</td>
<td>int</td>
<td>%td</td>
<td>Date of entry</td>
</tr>
<tr>
<td>dox</td>
<td>int</td>
<td>%td</td>
<td>Date of exit</td>
</tr>
<tr>
<td>dob</td>
<td>int</td>
<td>%td</td>
<td>Date of birth</td>
</tr>
</tbody>
</table>

Sorted by: id

. notes

See Structural models 7: Survival models in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Censoring and truncation
- Using stset to declare survival characteristics
- Fitting the loglogistic model
- Fitting the model with the Builder
Censoring and truncation

Survival datasets often include observations that are right-censored or left-truncated or both. When fitting survival models with family(exponential), family(gamma), family(loglogistic), family(lognormal), and family(weibull), we can include the suboption ltruncated(), which specifies a left-truncation point, and the suboption failure(), which indicates whether an observation records a failure or whether it was censored. For instance,

```
.gsem failtime ..., failure(weibull, failure(failed) ltruncated(enter))
```

fits a Weibull model for time to failure (failtime) where a failure is observed for observations with failed = 1 and observations with failed = 0 are censored. In addition, observations are left-truncated at the time specified in enter.

Using stset to declare survival characteristics

If you are familiar with Stata’s other commands for survival analysis, such as streg, you may have been surprised to see that we did not need to stset our data to specify the failure time, censoring, and the truncation variables before using the gsem command above. Most survival analysis commands rely on stset to record information on censoring and truncation.

stset can also be used to transform time in an analysis-time metric. Analysis time is the time a subject is at risk. In this metric, a time of 0 is the time when the subject becomes at risk. gsem assumes that the dependent variable is already recorded in analysis time. If you have data in another scale, such as calendar time, you will need to transform your variables.

Although not required by gsem, stset provides a convenient way to transform data into analysis time. You can also specify truncation and censoring variables just as you would before fitting survival models with other commands. See [ST] stset for details on declaring survival data using this command.

In the dataset described above, dox records the date an individual is diagnosed with coronary heart disease, cancer, or another disease of interest. fail has a nonzero code for individuals diagnosed with a disease and a zero for individuals who were censored. dob records date of birth, and doe is the date of entry to the study. We could transform the data into analysis time using stset as follows:

```
.stset dox, failure(fail) origin(time dob) enter(time doe) id(id)
```

This syntax gives us analysis time in days. Instead, we want to express the analysis time in years, so we type

```
.stset dox, failure(fail) origin(time dob) enter(time doe) id(id) scale(365.25)
```

<table>
<thead>
<tr>
<th>id: id</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure event: fail != 0 &amp; fail &lt; .</td>
</tr>
<tr>
<td>obs. time interval: (dox[_n-1], dox]</td>
</tr>
<tr>
<td>enter on or after: time doe</td>
</tr>
<tr>
<td>exit on or before: failure</td>
</tr>
<tr>
<td>t for analysis: (time-origin)/365.25</td>
</tr>
<tr>
<td>origin: time dob</td>
</tr>
</tbody>
</table>

337 total observations
0 exclusions
337 observations remaining, representing 337 subjects
80 failures in single-failure-per-subject data
4,603.669 total analysis time at risk and under observation
at risk from t = 0
earliest observed entry t = 30.07529
last observed exit t = 69.99863
The `stset` command generates the variables `_t0`, `_t`, `_d`, and `_st`.

Variable `_st` is equal to 1 unless there is a problem in the settings (for example, somebody dies before being born), in which case it is equal to 0.

Variable `_t0` indicates when the individuals enter the study, in the analysis-time scale. Variable `_t` indicates when the individual failed or was censored, also in analysis-time scale. Variable `_d` is the failure indicator.

For example,

```
. list dob doe dox fail _t0 _t _d _st if id == 1
```

<table>
<thead>
<tr>
<th></th>
<th>dob</th>
<th>doe</th>
<th>dox</th>
<th>fail</th>
<th>_t0</th>
<th>_t</th>
<th>_d</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>04jan1915</td>
<td>16aug1964</td>
<td>01dec1976</td>
<td>0</td>
<td>49.615332</td>
<td>61.908282</td>
<td>0</td>
</tr>
</tbody>
</table>

This participant was born in 1915, entered the study in 1964, and was censored in 1976. In analysis time, this is expressed as follows: The person entered the study at age 49.6 and was censored at age 61.9.

**Fitting the loglogistic model**

We can use the variables created by `stset` to fit the model
We specify these variables directly in our `gsem` command.

```stata
.gsem (_t <- energy job height weight,
    > family(loglogistic, failure(_d) ltruncated(_t0)))
```

Generalized structural equation model  Number of obs = 332
Response : _t  No. of failures = 78
Family : loglogistic  Time at risk = 4533.5715
Form : accelerated failure-time
Link : log
Log likelihood = -378.82795

| Coef.  | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|--------|-----------|------|------|----------------------|
| _t     |           |      |      |                      |
| energy | 0.0694548 | 0.0583516 | 1.19 | 0.234                | -.0449123, 0.1838219 |
| job    | 0.0102962 | 0.0297246 | 0.35 | 0.729                | -.0479629, 0.0685552 |
| height | 0.0107453 | 0.0047026 | 2.28 | 0.022                | 0.0015283, 0.0199623 |
| weight | 0.0004624 | 0.0025579 | 0.18 | 0.857                | -.004551, 0.0054758  |
| _cons  | 2.210313  | 0.7830543 | 2.82 | 0.005                | 0.675544, 3.745071   |
| _t     |           |      |      |                      |
| logs   | -1.818008 | 0.1701509 | -2.151498 | -1.484518 |

This model is fit in the accelerated failure-time metric, and `gsem` reports coefficients. We can use `estat eform` to obtain exponentiated coefficients, which are interpreted as time ratios.

```stata
.estat eform
```

| exp(b)  | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|---------|-----------|------|------|----------------------|
| _t      |           |      |      |                      |
| energy  | 1.071924  | 0.0625485 | 1.19 | 0.234                | 0.9560813, 1.201802  |
| job     | 1.010349  | 0.0300322 | 0.35 | 0.729                | 0.9531691, 1.07096   |
| height  | 1.010803  | 0.0047534 | 2.28 | 0.022                | 1.000153, 1.020163   |
| weight  | 1.000463  | 0.0025591 | 0.18 | 0.857                | 0.9954593, 1.005491  |
| _cons   | 9.118567  | 7.140333 | 2.82 | 0.005                | 1.965122, 42.31201   |

Each of the time ratios is just above 1, so an increase in any of the covariates would slightly increase the expected time to failure (for example, having a job increases the expected time until developing a disease by 1.01). However, only the time ratio for height is significantly different from 1.

### Fitting the model with the Builder

Use the diagram in *Fitting the loglogistic model* above for reference.

1. Open the dataset.
   In the Command window, type
   ```stata
   . use https://www.stata-press.com/data/r16/gsem_diet
   . stset dox, failure(fail) origin(time dob) enter(time doe) id(id) scale(365.25)
   ```

2. Open a new Builder diagram.
   Select menu item *Statistics > SEM (structural equation modeling) > Model building and estimation.*
3. Put the Builder in \texttt{gsem} mode by clicking on the $\text{\$}$ button.

4. Create the loglogistic regression component for $\_t$.

Select the Add regression component tool, $\text{\$}$, and then click in the diagram about one-third of the way in from the left and halfway down.

In the resulting dialog box,

a. select $\_t$ in the \textit{Dependent variable} control;

b. check \textit{Make response generalized};

c. select Loglogistic, Log in the \textit{Family/Link} control;

d. select the \textit{Select variables} radio button (it may already be selected);

e. use the \textit{Independent variables} control to select the variables energy, job, height, and weight;

f. select Left in the \textit{Independent variables’ direction} control;

g. click on \text{OK}.

If you wish, move the component by clicking on any variable and dragging it.

5. Specify censoring and truncation variables.

a. Choose the Select tool, $\checkmark$.

b. Click on the box for $\_t$.

c. In the Contextual Toolbar, click on the $\text{Properties...}$ button.

d. In the resulting \textit{Variable properties} dialog box, click on the $\text{Failure and truncation...}$ button in the \textit{Variable} tab.

e. In the resulting \textit{Failure and truncation} dialog box, select $\_d$ in the \textit{Failure variable} control. Check the \textit{Survival time is left-truncated} box. Select the \textit{Variable containing truncation values} radio button, and select $\_t0$ in the \textit{Variable} control. Click on \text{OK}.

f. Click on \text{OK} in the \textit{Variables properties} dialog box.

6. Clean up.

The box for $\_t$ is created closer to the independent variables than it is in the example diagram. Use the Select tool, $\checkmark$, and click on the box for $\_t$. Drag it to the right to allow more space for results along the paths.

7. Estimate.

Click on the $\text{Estimate}$ button, $\text{\$}$, in the Standard Toolbar, and then click on $\text{OK}$ in the resulting $\text{GSEM estimation options}$ dialog box.

You can open a completed diagram in the Builder by typing

$\text{webgetsem gsem_llog}$

\textbf{Reference}

Also see

[SEM] Example 47g — Exponential survival model
[SEM] Example 49g — Multiple-group Weibull survival model
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
[SEM] estat eform — Display exponentiated coefficients
Example 49g — Multiple-group Weibull survival model

Description

Below we demonstrate gsem’s group() option, which allows us to fit models in which coefficients, intercepts, and other types of parameters differ across groups of the data. We will fit a Weibull model to the following survival data

```stata
use https://www.stata-press.com/data/r16/gsem_cancer
(Patient Survival in Drug Trial)
```

```stata
describe
Contains data from https://www.stata-press.com/data/r16/gsem_cancer.dta
obs: 48 Patient Survival in Drug Trial
vars: 4 16 Jan 2019 15:41
(_dta has notes)
```

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage type</th>
<th>display format</th>
<th>value label</th>
</tr>
</thead>
<tbody>
<tr>
<td>studytime</td>
<td>byte</td>
<td>%8.0g</td>
<td>Months to death or end of exp.</td>
</tr>
<tr>
<td>died</td>
<td>byte</td>
<td>%8.0g</td>
<td>1 if patient died</td>
</tr>
<tr>
<td>drug</td>
<td>byte</td>
<td>%8.0g</td>
<td>Drug type (1=placebo)</td>
</tr>
<tr>
<td>age</td>
<td>byte</td>
<td>%8.0g</td>
<td>Patient's age at start of exp.</td>
</tr>
</tbody>
</table>

Sorted by:

```stata
notes
_dta:
  1. Artificial data on cancer patient survival.
  2. Drug 1 is a placebo. Drugs 2 and 3 are alternative treatments.
```

See [SEM] Example 47g and [SEM] Example 48g for how to fit survival models using gsem. In this example, we focus on fitting multiple-group models using gsem’s group() and ginvariant() options. See [SEM] Intro 6 for background on these options.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the multiple-group model
- Fitting the model with the Builder
Fitting the multiple-group model

We want to fit the model

\[
\text{studytime} \sim \text{Weibull}(\log(\text{age}))
\]

If we do not allow for group differences, we can fit this model by typing

```
.gsem (studytime <- age, family(weibull, failure(died)))
```

However, this study used one placebo and two drugs. We want to allow the intercept and auxiliary parameter to vary across the levels of `drug` but constrain the coefficient on `age` to be equal across the levels of `drug`. In other words, we want to fit a stratified Weibull model. We add the `group(drug)` and `ginvariant(coef)` options to our `gsem` command to specify that `drug` is the group identifier and that coefficients should not vary across groups.

```
.gsem (studytime <- age, family(weibull, failure(died))),  
> group(drug) ginvariant(coef)
```

<table>
<thead>
<tr>
<th>Group</th>
<th>Number of obs = 20</th>
<th>Number of groups = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>studytime</td>
<td>No. of failures = 19</td>
</tr>
<tr>
<td>Family</td>
<td>Weibull</td>
<td>Time at risk = 180</td>
</tr>
<tr>
<td>Form</td>
<td>proportional hazards</td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td>log</td>
<td></td>
</tr>
</tbody>
</table>

| studytime | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|-----------|-------|-----------|---|-------|------------------|
| age | 0.1212332 | 0.0367538 | 3.30 | 0.001 | 0.049197 - 0.1932694 |
| _cons | -10.36921 | 2.341022 | -4.43 | 0.000 | -14.95753 - 5.780896 |
| /studytime | ln_p | 0.4541282 | 0.1715663 | 0.1178645 - 0.7903919 |

<table>
<thead>
<tr>
<th>Group</th>
<th>Number of obs = 14</th>
<th>Number of groups = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>studytime</td>
<td>No. of failures = 6</td>
</tr>
<tr>
<td>Family</td>
<td>Weibull</td>
<td>Time at risk = 209</td>
</tr>
<tr>
<td>Form</td>
<td>proportional hazards</td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td>log</td>
<td></td>
</tr>
</tbody>
</table>

| studytime | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|-----------|-------|-----------|---|-------|------------------|
| age | 0.1212332 | 0.0367538 | 3.30 | 0.001 | 0.049197 - 0.1932694 |
| _cons | -14.93039 | 3.445179 | -4.33 | 0.000 | -21.68282 - 8.177965 |
| /studytime | ln_p | 0.9413477 | 0.2943728 | 0.3643876 - 1.518308 |
Example 49g — Multiple-group Weibull survival model

<table>
<thead>
<tr>
<th>Group</th>
<th>Number of obs = 14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>studytime</td>
</tr>
<tr>
<td>Family</td>
<td>Weibull</td>
</tr>
<tr>
<td>Form</td>
<td>proportional hazards</td>
</tr>
<tr>
<td>Link</td>
<td>log</td>
</tr>
</tbody>
</table>

| studytime | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|-----------|-------|-----------|---|------|---------------------|
| age       | .1212332 | .0367538 | 3.30 | 0.001 | .049197 — .1932694 |
| _cons     | -14.08495 | 3.242463 | -4.34 | 0.000 | -20.44006 — -7.72984 |

| /studytime | ln_p | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|------------|------|-------|-----------|---|------|---------------------|
| ln_p       | .6735495 | .369625 | -.0509022 | 1.398001 | |

Notes:

1. In [SEM] Intro 6, we wrote that gsem classifies each parameter into one of seven classes, which are the following:

<table>
<thead>
<tr>
<th>Class description</th>
<th>Class name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. intercepts and cutpoints</td>
<td>cons</td>
</tr>
<tr>
<td>2. fixed coefficients</td>
<td>coef</td>
</tr>
<tr>
<td>3. latent variable coefficients</td>
<td>loading</td>
</tr>
<tr>
<td>4. covariances of errors</td>
<td>errvar</td>
</tr>
<tr>
<td>5. scaling parameters</td>
<td>scale</td>
</tr>
<tr>
<td>6. means of exogenous variables</td>
<td>means</td>
</tr>
<tr>
<td>7. covariances of exogenous latent variables</td>
<td>covex</td>
</tr>
<tr>
<td>8. all the above</td>
<td>all</td>
</tr>
<tr>
<td>9. none of the above</td>
<td>none</td>
</tr>
</tbody>
</table>

By default, classes 1, 2, and 3 are constrained to be equal, and the others are allowed to vary.

2. In the output, we have a separate table of parameter estimates for each level of drug. The coefficient on age is 0.12 in all groups, but, as expected, the other parameters differ across groups.

We can replay the results with the byparm option to request that results be sorted by parameter rather than by groups. This output makes it easy to compare estimates across the groups. Alternatively, we could have added the byparm option when we fit the model.

```
.gsem, byparm
Generalized structural equation model
Grouping variable = drug
Group 1
Number of obs = 20
Response : studytime
Number of groups = 3
Family : Weibull
Time at risk = 180
Form : proportional hazards
Link : log
```

```
2
Group 2
Number of obs = 14
Response : studytime
Number of obs = 6
Family : Weibull
Time at risk = 209
Form : proportional hazards
Link : log
```
Group : 3  Number of obs = 14
Response : studytime  No. of failures = 6
Family : Weibull  Time at risk = 355
Form : proportional hazards
Link : log
Log likelihood = -109.28976

( 1) [studytime]1bn.drug#c.age - [studytime]3.drug#c.age = 0
( 2) [studytime]2.drug#c.age - [studytime]3.drug#c.age = 0

Coef. Std. Err.  z   P>|z|  [95% Conf. Interval]
studytime
   age
      1  .1212332  .0367538  3.30  0.001  .049197  .1932694
      2  .1212332  .0367538  3.30  0.001  .049197  .1932694
      3  .1212332  .0367538  3.30  0.001  .049197  .1932694
   _cons
      1 -10.36921  2.341022 -4.43  0.000 -14.95753 -5.780896
      2 -14.93039  3.445179 -4.33  0.000 -21.68282 -8.177965
      3 -14.08495  3.242463 -4.34  0.000 -20.44006 -7.72984
/studytime
   ln_p
      1  .4541282  .1715663  11.78645  .7903919
      2  .9413477  .2943728  3.24263  1.398001
      3  .6735495  .369625  -.0509022  1.398001

The estimated intercepts, labeled _cons in the first section of the table, do not vary much across groups. We can perform a Wald test of whether these coefficients are the same using the test command. If we replay the results by typing gsem, coeflegend, we find that we can refer to the three intercepts as _b[studytime:1.drug], _b[studytime:2.drug], and _b[studytime:3.drug]. Therefore, our test command is

```
( 1) [studytime]1bn.drug - [studytime]2.drug = 0
( 2) [studytime]1bn.drug - [studytime]3.drug = 0
       chi2(  2) =     5.49
       Prob > chi2 =   0.0641
```

Using a 5% significance level, we fail to reject the null hypothesis that the intercepts are equal. If we wanted to refit the model based on these findings, we could include cons in the ginvariant() option as follows and constrain both the coefficients and the intercepts across groups.

```
. gsem (studytime <- age, family(weibull, failure(died))), ///
group(drug) ginvariant(coef cons)
```

**Fitting the model with the Builder**

Use the diagram in *Fitting the multiple-group model* above for reference.

1. Open the dataset.
   In the Command window, type

```
   . use https://www.stata-press.com/data/r16/gsem_cancer
```
2. Open a new Builder diagram.
   Select menu item **Statistics > SEM (structural equation modeling) > Model building and estimation**.

3. Put the Builder in `gsem` mode by clicking on the `g` button.

4. Create the Weibull regression component for `studytime`.
   Select the Add regression component tool, `.Controls`, and then click in the diagram about one-third of the way in from the left and halfway down.
   In the resulting dialog box,
   a. select `studytime` in the *Dependent variable* control;
   b. check *Make response generalized*;
   c. select Weibull, Log in the *Family/Link* control;
   d. select the *Select variables* radio button (it may already be selected);
   e. use the *Independent variables* control to select the variable `age`;
   f. select *Left* in the *Independent variables’ direction* control;
   g. click on **OK**.
   If you wish, move the component by clicking on any variable and dragging it.

5. Specify the censoring variable.
   a. Choose the Select tool, `.
   b. Click on the box for `studytime`.
   c. In the Contextual Toolbar, click on the `Properties...` button.
   d. In the resulting *Variable properties* dialog box, click on the `Failure and truncation...` button in the *Variable* tab.
   e. In the resulting *Failure and truncation* dialog box, select `died` in the *Failure variable* control. Click on **OK**.
   f. Click on **OK** in the *Variables properties* dialog box.

6. Clean up.
   The box for `_t` is created closer to the independent variables than it is in the example diagram.
   Use the Select tool, ` `, and click on the box for `_t`. Drag it to the right to allow more space for results along the paths.

7. Estimate.
   Click on the **Estimate** button, ` `, in the Standard Toolbar.
   In the resulting dialog box, do the following:
   a. Select the *Group* tab.
   b. Select the *Group analysis* radio button. Select the variable `drug` in the *Group variable* control.
   c. Select *Fixed coefficients* in the *Parameters that are equal across groups* control.
   d. Click on **OK**.
e. In the Standard Toolbar, use the Group control to toggle between results for group 1 and group 2.

You can open a completed diagram in the Builder by typing

`.webgetsem gsem.grp`

Also see

[SEM] Example 47g — Exponential survival model
[SEM] Example 48g — Loglogistic survival model with censored and truncated data
[SEM] Intro 5 — Tour of models
[SEM] Intro 6 — Comparing groups
[SEM] gsem — Generalized structural equation model estimation command
Example 50g — Latent class model

Description

To demonstrate latent class models, we use the following data:

```stata
use https://www.stata-press.com/data/r16/gsem_lca1
(Latent class analysis)
```

Contains data from https://www.stata-press.com/data/r16/gsem_lca1.dta

| obs: 216 | Latent class analysis |
| vars: 4  | 17 Jan 2019 12:52     |

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value label</th>
</tr>
</thead>
<tbody>
<tr>
<td>accident</td>
<td>byte</td>
<td>%9.0g</td>
<td>would testify against friend in accident case</td>
</tr>
<tr>
<td>play</td>
<td>byte</td>
<td>%9.0g</td>
<td>would give negative review of friend's play</td>
</tr>
<tr>
<td>insurance</td>
<td>byte</td>
<td>%9.0g</td>
<td>would disclose health concerns to friend's play</td>
</tr>
<tr>
<td>stock</td>
<td>byte</td>
<td>%9.0g</td>
<td>would keep company secret from friend</td>
</tr>
</tbody>
</table>

Sorted by: accident play insurance stock

. notes in 1/4

_dta:

1. Data from Samuel A. Stouffer and Jackson Toby, March 1951, "Role conflict and personality", _The American Journal of Sociology_, vol. 56 no. 5, 395-406.
2. Variables represent responses of students from Harvard and Radcliffe who were asked how they would respond to four situations. Respondents selected either a particularistic response (based on obligations to a friend) or universalistic response (based on obligations to society).
3. Each variable is coded with 0 indicating a particularistic response and 1 indicating a universalistic response.
4. For a full description of the questions, type "notes in 5/8".

See _Latent class models_ in [SEM] Intro 5 for background.

Remarks and examples

A latent class model is characterized by having a categorical (rather than continuous) latent variable. The levels of the categorical latent variable represent groups in the population and are called classes. We are interested in identifying and understanding these classes.
Goodman (2002) fits a variety of latent class models to the dataset described above with a focus on understanding how groups of individuals differ in response to situations that require making a decision between helping a friend (a particularistic choice) and doing what is right for society (a universalistic choice). Individuals were asked how they would respond in four such situations, and their responses were recorded in the variables accident, play, insurance, and stock. These variables are coded such that 1 is a universalistic response and 0 is a particularistic response.

To fit a latent class model, we must specify the number of classes in the latent variable. In the basic form of the latent class model demonstrated here, we have one categorical latent variable with two classes. The parameters in the model, namely, the intercepts in logistic regression models for the four observed variables, are allowed to vary across the classes.

More specifically, the model that we will fit estimates an intercept, $\alpha$, for each observed variable for the first class,

$$
\Pr(\text{accident} = 1 | C = 1) = \frac{\exp(\alpha_{11})}{1 + \exp(\alpha_{11})}
$$

$$
\Pr(\text{play} = 1 | C = 1) = \frac{\exp(\alpha_{21})}{1 + \exp(\alpha_{21})}
$$

$$
\Pr(\text{insurance} = 1 | C = 1) = \frac{\exp(\alpha_{31})}{1 + \exp(\alpha_{31})}
$$

$$
\Pr(\text{stock} = 1 | C = 1) = \frac{\exp(\alpha_{41})}{1 + \exp(\alpha_{41})}
$$

and a corresponding intercept for the second class,

$$
\Pr(\text{accident} = 1 | C = 2) = \frac{\exp(\alpha_{12})}{1 + \exp(\alpha_{12})}
$$

$$
\Pr(\text{play} = 1 | C = 2) = \frac{\exp(\alpha_{22})}{1 + \exp(\alpha_{22})}
$$

$$
\Pr(\text{insurance} = 1 | C = 2) = \frac{\exp(\alpha_{32})}{1 + \exp(\alpha_{32})}
$$

$$
\Pr(\text{stock} = 1 | C = 2) = \frac{\exp(\alpha_{42})}{1 + \exp(\alpha_{42})}
$$

We also estimate the probability of being in each class using multinomial logistic regression,

$$
\Pr(C = 1) = \frac{e^{\gamma_1}}{e^{\gamma_1} + e^{\gamma_2}}
$$

$$
\Pr(C = 2) = \frac{e^{\gamma_2}}{e^{\gamma_1} + e^{\gamma_2}}
$$

where $\gamma_1$ and $\gamma_2$ are intercepts in the multinomial logit model. By default, the first class will be treated as the base, so $\gamma_1 = 0$. 
To fit this model, we type

```
gsem (accident play insurance stock <- ), logit lclass(C 2)
```

No variables are listed on the right side of the arrow because we are fitting intercept-only models for each observed variable. `logit` specifies that we are fitting logistic regression models for all four variables. The `lclass(C 2)` option specifies that the name of our categorical latent variable is C and that it has two latent classes.

The result of typing our estimation command is

```
gsem (accident play insurance stock <- ), logit lclass(C 2)
```

**Fitting class model:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(class) log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-149.71979</td>
</tr>
<tr>
<td>1</td>
<td>-149.71979</td>
</tr>
</tbody>
</table>

**Fitting outcome model:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(outcome) log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-403.97142</td>
</tr>
<tr>
<td>1</td>
<td>-398.15909</td>
</tr>
<tr>
<td>2</td>
<td>-397.81953</td>
</tr>
<tr>
<td>3</td>
<td>-397.8164</td>
</tr>
<tr>
<td>4</td>
<td>-397.8164</td>
</tr>
</tbody>
</table>

**Refining starting values:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(EM) log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-570.24204</td>
</tr>
<tr>
<td>1</td>
<td>-576.20485</td>
</tr>
<tr>
<td>2</td>
<td>-577.41464</td>
</tr>
<tr>
<td>3</td>
<td>-575.92422</td>
</tr>
<tr>
<td>4</td>
<td>-576.88554</td>
</tr>
<tr>
<td>5</td>
<td>-573.90567</td>
</tr>
<tr>
<td>6</td>
<td>-571.99868</td>
</tr>
<tr>
<td>7</td>
<td>-569.97482</td>
</tr>
<tr>
<td>8</td>
<td>-567.90955</td>
</tr>
<tr>
<td>9</td>
<td>-565.86392</td>
</tr>
<tr>
<td>10</td>
<td>-563.88815</td>
</tr>
<tr>
<td>11</td>
<td>-562.02165</td>
</tr>
<tr>
<td>12</td>
<td>-560.29231</td>
</tr>
<tr>
<td>13</td>
<td>-558.71641</td>
</tr>
<tr>
<td>14</td>
<td>-557.29974</td>
</tr>
<tr>
<td>15</td>
<td>-556.03949</td>
</tr>
<tr>
<td>16</td>
<td>-554.92679</td>
</tr>
<tr>
<td>17</td>
<td>-553.94914</td>
</tr>
<tr>
<td>18</td>
<td>-553.09241</td>
</tr>
<tr>
<td>19</td>
<td>-552.34233</td>
</tr>
<tr>
<td>20</td>
<td>-551.68539</td>
</tr>
</tbody>
</table>

Note: EM algorithm reached maximum iterations.

**Fitting full model:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-504.62913</td>
</tr>
<tr>
<td>1</td>
<td>-504.47255</td>
</tr>
<tr>
<td>2</td>
<td>-504.46773</td>
</tr>
<tr>
<td>3</td>
<td>-504.46767</td>
</tr>
<tr>
<td>4</td>
<td>-504.46767</td>
</tr>
</tbody>
</table>

Generalized structural equation model

| Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------|-----------|------|------|----------------------|
| 1.C   | (base outcome) |      |      |                      |
| 2.C _cons |             | -.9482041 | .2886333 | -3.29     | 0.001    | -1.513915 | -.3824933 |
Example 50g — Latent class model

Class : 1
Response : accident
Family : Bernoulli
Link : logit
Response : play
Family : Bernoulli
Link : logit
Response : insurance
Family : Bernoulli
Link : logit
Response : stock
Family : Bernoulli
Link : logit

|           | Coef.    | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-----------|----------|-----------|-------|------|---------------------|
| accident  | .9128742 | .1974695  | 4.62  | 0.000| .5258411 1.299907   |
| play      | -.7099072| .2249096  | -3.16 | 0.002| -1.150722 -.2690926 |
| insurance | -.6014307| .2123096  | -2.83 | 0.005| -1.01755 -.1853115  |
| stock     | -1.880142| .3337665  | -5.63 | 0.000| -2.534312 -1.225972 |

Class : 2
Response : accident
Family : Bernoulli
Link : logit
Response : play
Family : Bernoulli
Link : logit
Response : insurance
Family : Bernoulli
Link : logit
Response : stock
Family : Bernoulli
Link : logit

|           | Coef.    | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-----------|----------|-----------|-------|------|---------------------|
| accident  | 4.983017 | 3.745987  | 1.33  | 0.183| -2.358982 12.32502  |
| play      | 2.747366 | 1.165853  | 2.36  | 0.018| .4623372 5.032395   |
| insurance | 2.534582 | .9644841  | 2.63  | 0.009| .6442279 4.424936   |
| stock     | 1.203416 | .5361735  | 2.24  | 0.025| .1525356 2.254297   |
Notes:

1. The output shows four iteration logs. The first three are for models that are fit to obtain good starting values. Starting values are challenging for latent class models, and \texttt{gsem} provides a variety of options for specifying and computing starting values. See [SEM] \texttt{gsem estimation options} and [SEM] Intro 12 for more information on these options.

2. The first table in the output provides the estimated coefficients in the multinomial logit model for \( C \).

3. The next two tables are the results for the logistic regression models for the first and second classes.

To better understand this model, let’s examine how the probabilities of giving a universalistic response differ across classes. The \texttt{estat lcmean} command reports class-specific marginal means for each variable. Because we are using logistic regression, these means are actually the predicted probabilities.

\begin{verbatim}
. estat lcmean
Latent class marginal means
Number of obs = 216

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin</td>
<td>Std. Err.</td>
<td></td>
</tr>
<tr>
<td>1 accident</td>
<td>.7135879</td>
<td>.0403588</td>
<td>.6285126 .7858194</td>
</tr>
<tr>
<td>play</td>
<td>.3296193</td>
<td>.0496984</td>
<td>.2403573 .4331299</td>
</tr>
<tr>
<td>insurance</td>
<td>.3540164</td>
<td>.0485528</td>
<td>.2655049 .4538042</td>
</tr>
<tr>
<td>stock</td>
<td>.1323726</td>
<td>.0383331</td>
<td>.0734875 .2268872</td>
</tr>
<tr>
<td>2 accident</td>
<td>.9931933</td>
<td>.0253243</td>
<td>.0863544 .9999956</td>
</tr>
<tr>
<td>play</td>
<td>.9397644</td>
<td>.0659957</td>
<td>.6135685 .9935191</td>
</tr>
<tr>
<td>insurance</td>
<td>.9265309</td>
<td>.0656538</td>
<td>.6557086 .9881667</td>
</tr>
<tr>
<td>stock</td>
<td>.7691320</td>
<td>.0952072</td>
<td>.5380601 .9050206</td>
</tr>
</tbody>
</table>
\end{verbatim}

The first section of this table reports the probabilities for class 1. In this class, the probability of giving a universalistic response to the first question—the question that concerns testifying against a friend who was involved in an accident—is 0.714. The probability of giving a universalistic response to the last question—the question about warning a friend about a stock price that is about to fall—is 0.132.

The second section of the table reports the corresponding probabilities for class 2. We find that the probability of giving a universalistic response to each question is higher in class 2 than in class 1. Class 2 appears to represent a more universalistically inclined group.

We can estimate probabilities of being in each class using \texttt{estat lcprob}.

\begin{verbatim}
. estat lcprob
Latent class marginal probabilities
Number of obs = 216

<table>
<thead>
<tr>
<th>C</th>
<th>Delta-method</th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.7207539</td>
<td>.0580926</td>
<td>.5944743 .8196407</td>
</tr>
<tr>
<td>2</td>
<td>.2792461</td>
<td>.0680926</td>
<td>.1803593 .4055257</td>
</tr>
</tbody>
</table>
\end{verbatim}
This indicates that 72% of individuals are expected to be in the less universalistic class and 28% are expected to be in the more universalistic class.

We can use the predictions of the posterior probability of class membership to evaluate an individual’s probability of being in each class.

```
.predict classpost*, classposteriorpr
.list in 1, abbrev(10)
```

<table>
<thead>
<tr>
<th>accident</th>
<th>play</th>
<th>insurance</th>
<th>stock</th>
<th>classpost1</th>
<th>classpost2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>.999975</td>
<td>.000025</td>
</tr>
</tbody>
</table>

For the first individual in our dataset, who responded with a particularistic answer to all four questions, the probability of being in class 1, the less universalistic class, is almost 1.

We can determine the expected class for each individual based on whether the posterior probability is greater than 0.5.

```
.generate expclass = 1 + (classpost2>0.5)
.tabulate expclass
```

<table>
<thead>
<tr>
<th>expclass</th>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>145</td>
<td>67.13</td>
<td>67.13</td>
</tr>
<tr>
<td>2</td>
<td>71</td>
<td>32.87</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Total | 216   | 100.00  |

In our dataset, 145 individuals are expected to be in class 1 and 71 individuals are expected to be in class 2.

References


Also see

[SEM] Example 51g — Latent class goodness-of-fit statistics
[SEM] Example 52g — Latent profile model
[SEM] Intro 5 — Tour of models
[SEM] gsem — Generalized structural equation model estimation command
[SEM] estat lcmean — Latent class marginal means
[SEM] estat lcprob — Latent class marginal probabilities
Example 51g — Latent class goodness-of-fit statistics

Description

Here we demonstrate how to obtain goodness-of-fit statistics for latent class models.

We continue with [SEM] Example 50g, where we fit a two-class model:

```
. use https://www.stata-press.com/data/r16/gsem_lca1
. gsem (accident play insurance stock <- ), logit lclass(C 2)
```

See Latent class models in [SEM] Intro 5 for background.

Remarks and examples

Remarks are presented under the following headings:

- Likelihood-ratio ($G^2$) test
- Comparing models

Likelihood-ratio ($G^2$) test

For standard latent class models with observed variables that are all categorical, one way to evaluate model fit is to compare the model we have just fit with a saturated model. We can use the `estat lcgof` command to perform a likelihood-ratio test of whether our model fits as well as the saturated model. The corresponding likelihood-ratio statistic is sometimes referred to as $G^2$ in latent class analysis literature.

```
. estat lcgof
```

<table>
<thead>
<tr>
<th>Fit statistic</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chi2_ms(6)</td>
<td>2.720</td>
<td>model vs. saturated</td>
</tr>
<tr>
<td>p &gt; chi2</td>
<td>0.843</td>
<td></td>
</tr>
<tr>
<td>Information criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIC</td>
<td>1026.935</td>
<td>Akaike’s information criterion</td>
</tr>
<tr>
<td>BIC</td>
<td>1057.313</td>
<td>Schwarz’s Bayesian information criterion</td>
</tr>
</tbody>
</table>

We fail to reject the null hypothesis that our model fits as well as the saturated model.

`estat lcgof` also reports Akaike’s information criterion (AIC) and Schwarz’s Bayesian information criterion (BIC). These are useful for comparing models but not useful for determining goodness-of-fit for a single model.
Comparing models

In latent class analysis, we often compare models that have different numbers of classes. Following Goodman (2002), we compare models that allow for one, two, and three latent classes. We have already fit the two-class model using the \texttt{gsem} command above. Before we move on, we will store the results of this model.

\begin{verbatim}
. estimates store twoclass
\end{verbatim}

Next, we fit the one-class model, store the results, and perform the likelihood-ratio test comparing it with the saturated model.

\begin{verbatim}
. quietly gsem (accident play insurance stock <- ), logit lclass(C 1)
. estimates store oneclass
. estat lcgof
\end{verbatim}

<table>
<thead>
<tr>
<th>Fit statistic</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chi2_ms(11)</td>
<td>81.084</td>
<td>model vs. saturated</td>
</tr>
<tr>
<td>p &gt; chi2</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>Information criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIC</td>
<td>1095.300</td>
<td>Akaike’s information criterion</td>
</tr>
<tr>
<td>BIC</td>
<td>1108.801</td>
<td>Bayesian information criterion</td>
</tr>
</tbody>
</table>

We reject the null hypothesis in this case. The one-class model does not fit well.

We also fit the three-class model.

\begin{verbatim}
. quietly gsem (accident play insurance stock <- ), logit lclass(C 3)
. estimates store threeclass
. estat lcgof
\end{verbatim}

<table>
<thead>
<tr>
<th>Fit statistic</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chi2_ms(1)</td>
<td>0.387</td>
<td>model vs. saturated</td>
</tr>
<tr>
<td>p &gt; chi2</td>
<td>0.534</td>
<td></td>
</tr>
<tr>
<td>Information criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIC</td>
<td>1034.602</td>
<td>Akaike’s information criterion</td>
</tr>
<tr>
<td>BIC</td>
<td>1081.856</td>
<td>Bayesian information criterion</td>
</tr>
</tbody>
</table>

Based on this test, the three-class model, like the two-class model, does not fit worse than the saturated model.

We will compare our three models using \texttt{AIC} and \texttt{BIC}. Smaller values of \texttt{AIC} and \texttt{BIC} are better. We could look back at the \texttt{AIC} and \texttt{BIC} values reported by our three \texttt{estat lcgof} commands, but we will instead create a table that reports these information criteria for all three models using \texttt{estimates stats}. 

\begin{table}[h]
\centering
\begin{tabular}{ll}
\hline
Fit statistic & Value & Description &
\hline
Likelihood ratio & \multicolumn{2}{l}{Value} & Description \hline
\multicolumn{1}{l}{chi2_ms(11)} & 81.084 & model vs. saturated \hline
\multicolumn{1}{l}{p > chi2} & 0.000 &                       \hline
Information criteria & \multicolumn{2}{l}{Value} & Description \hline
AIC & 1095.300 & Akaike’s information criterion \hline
BIC & 1108.801 & Bayesian information criterion \hline
\end{tabular}
\caption{Latent class goodness-of-fit statistics}
\end{table}
Akaike’s information criterion and Bayesian information criterion

<table>
<thead>
<tr>
<th>Model</th>
<th>N</th>
<th>ll(null)</th>
<th>ll(model)</th>
<th>df</th>
<th>AIC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>oneclass</td>
<td>216</td>
<td>-543.6498</td>
<td>1105.3</td>
<td>4</td>
<td>1095.3</td>
<td>1108.801</td>
</tr>
<tr>
<td>twoclass</td>
<td>216</td>
<td>-504.4677</td>
<td>1026.935</td>
<td>9</td>
<td>1026.935</td>
<td>1057.313</td>
</tr>
<tr>
<td>threeclass</td>
<td>216</td>
<td>-503.3011</td>
<td>1034.602</td>
<td>14</td>
<td>1034.602</td>
<td>1081.856</td>
</tr>
</tbody>
</table>

Note: BIC uses N = number of observations. See [R] BIC note.

The two-class model has both the smallest AIC and the smallest BIC.

Reference


Also see

- [SEM] Example 50g — Latent class model
- [SEM] Intro 5 — Tour of models
- [SEM] gsem — Generalized structural equation model estimation command
- [SEM] estat lcgof — Latent class goodness-of-fit statistics
Example 52g — Latent profile model

Description

To demonstrate latent profile models, we use the following data:

```
. use https://www.stata-press.com/data/r16/gsem_lca2
(Latent profile analysis)
. describe
Contains data from https://www.stata-press.com/data/r16/gsem_lca2.dta
   obs:        145  Latent profile analysis
   vars:       7  18 Jan 2019 12:39
   (_dta has notes)

storage  display value
variable name   type format label

patient    int  %9.0g    Patient ID
relwgt     float %9.0g   Relative weight
fglucose   int  %9.0g    Fasting plasma glucose
glucose    float %9.0g   Glucose area (mg/10mL/hr)
insulin    float %9.0g   Insulin area (mIU/10mL/hr)
sspg       float %9.0g   Steady-state plasma glucose
cclass     byte  %17.0g  class    Clinical classification
```

Sorted by:
. notes
_dta:
3. Data includes variables related to diabetes for 145 non-obese adults.

See _Latent class models_ in [SEM] _Intro 5_ for background.

Remarks and examples

Remarks are presented under the following headings:

- Fitting the two-class model
- Comparing models
- Fitting the three-class model with covariances
Fitting the two-class model

In this manual, when we talk about latent class analysis, we are referring to an analysis that involves fitting models with categorical latent variables. Sometimes, these models are given more specific names. In [SEM] Example 50g, we fit a latent class model with a categorical latent variable and categorical observed variables. This is a typical latent class model. However, models with categorical latent variables are not limited to having categorical observed variables. A latent class model that instead has continuous observed variables is often referred to as a latent profile model.

Masyn (2013) uses the data described above to fit a series of latent profile models, each having one categorical latent variable and three observed variables, glucose, insulin, and sspg. The goal is to determine categories of diabetes based on these three variables. We begin by fitting a model in which the latent variable, C, has two classes. We fit a linear regression model for each observed variable where the intercept, $\alpha_{jc}$, is allowed to vary across the classes of the latent variable. Because we are using linear regression, we also estimate the variances of the error terms $e_{\text{glucose}}$, $e_{\text{insulin}}$, and $e_{\text{sspg}}$.

More specifically, for class 1 we fit

$$\begin{align*}
glucose &= \alpha_{11} + e_{\text{glucose}} \\
insulin &= \alpha_{21} + e_{\text{insulin}} \\
sspg &= \alpha_{31} + e_{\text{sspg}}
\end{align*}$$

and for class 2 we fit

$$\begin{align*}
glucose &= \alpha_{12} + e_{\text{glucose}} \\
insulin &= \alpha_{22} + e_{\text{insulin}} \\
sspg &= \alpha_{32} + e_{\text{sspg}}
\end{align*}$$

We also estimate the probability of being in each class using multinomial logistic regression,

$$\begin{align*}
\Pr(C = 1) &= \frac{e^{\gamma_1}}{e^{\gamma_1} + e^{\gamma_2}} \\
\Pr(C = 2) &= \frac{e^{\gamma_2}}{e^{\gamma_1} + e^{\gamma_2}}
\end{align*}$$

where $\gamma_1$ and $\gamma_2$ are intercepts in the multinomial logit model. By default, the first class will be treated as the base, so $\gamma_1 = 0$.

We will assume that the errors are uncorrelated, which is the default, and that the variances do not differ across classes, also the default.
. gsem (glucose insulin sspg <- _cons), lclass(C 2)

*iteration log omitted*

Generalized structural equation model  Number of obs = 145  
Log likelihood = -1702.5542

( 1) [\(\sigma^2\)gvar(e.glucose)\#1bn.C - \(\sigma^2\)gvar(e.glucose)\#2.C = 0]  
( 2) [\(\sigma^2\)gvar(e.insulin)\#1bn.C - \(\sigma^2\)gvar(e.insulin)\#2.C = 0]  
( 3) [\(\sigma^2\)gvar(e.sspg)\#1bn.C - \(\sigma^2\)gvar(e.sspg)\#2.C = 0]

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|------|----------------------|
| 1.C   | (base outcome) |       |      |                      |
| 2.C   | _cons     | -1.541025 | .2205682 | -6.99     | 0.000 | -1.973331 | -1.10872 |

Class : 1  
Response : glucose  
Family : Gaussian  
Link : identity  
Response : insulin  
Family : Gaussian  
Link : identity  
Response : sspg  
Family : Gaussian  
Link : identity

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|------|----------------------|
| glucose | _cons     | 41.22237 | 1.298051 | 31.76     | 0.000 | 38.67824 | 43.7665 |
| insulin | _cons     | 20.98005 | 1.000974 | 20.96     | 0.000 | 19.01817 | 22.94192 |
| sspg   | _cons     | 14.96579 | .6868081 | 21.79     | 0.000 | 13.61967 | 16.31191 |
| var(e.glucose) |       | 191.5596 | 23.83815 | 8.099200  | 0.000 | 150.0992| 244.4723 |
| var(e.insulin) |      | 119.0542 | 14.00336 | 8.503095  | 0.000 | 94.54204| 149.9217 |
| var(e.sspg)   |       | 55.91283 | 6.713667 | 8.363101  | 0.000 | 44.18801| 70.7487 |

* Example 52g — Latent profile model *
Example 52g — Latent profile model

Class: 2
Response: glucose
Family: Gaussian
Link: identity
Response: insulin
Family: Gaussian
Link: identity
Response: sspg
Family: Gaussian
Link: identity

|             | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------------|--------|-----------|-------|------|---------------------|
| glucose     | _cons  | 115.7123  | 2.849914 | 40.60 | 0.000  | 110.1266  | 121.2981 |
| insulin     | _cons  | 7.553144  | 2.160949 | 3.50  | 0.000  | 3.317761  | 11.78853 |
| sspg        | _cons  | 34.5529   | 1.53117 | 22.57 | 0.000  | 31.55187  | 37.55394 |

| var(e.gluc-e) | 191.5596 | 23.83815 | 150.0992 | 244.4723 |
| var(e.insu-n) | 119.0542 | 14.00336 | 94.54204 | 149.9217 |
| var(e.sspg)   | 55.91283 | 6.713667 | 44.18801 | 70.7487 |

. estimates store c2inv

Notes:
1. The first table in the output provides the estimated coefficients in the multinomial logit model for C.
2. The next two tables are the results for the linear regression models for the first and second classes.

Comparing models

Before we interpret any results, we will fit and compare other models. We modify our command above to specify that C has three, four, and then five latent classes, and we store the results of those models by typing

. gsem (glucose insulin sspg <- _cons), lclass(C 3)
. estimates store c3inv
. gsem (glucose insulin sspg <- _cons), lclass(C 4) ///
  startvalues(randomid, draws(5) seed(15)) emopts(iter(20))
. estimates store c4inv
. gsem (glucose insulin sspg <- _cons), lclass(C 5) ///
  startvalues(randomid, draws(5) seed(15)) emopts(iter(20))
. estimates store c5inv

For the models with four and five latent classes, we added the startvalues(randomid), draws(5) seed(15)) option to request that starting values be computed using random class assignments. In this option, draws(5) specifies that five random draws be taken and that the one with the best log likelihood after the EM iterations be selected. The emopts(iter(20)) option says that 20 EM iterations are used for each random draw. We also set the seed for reproducible results. We could have used the same options in the models with two classes and three classes. Difficulty finding good starting values is fairly common when fitting latent class models, so gsem provides a variety
of options for obtaining starting values. See [SEM] Intro 12 and [SEM] gsem estimation options for more information on starting values.

We can compare the four models fit above using Akaike’s information criterion (AIC) and Schwarz’s Bayesian information criterion (BIC).

```
. estimates stats c2inv c3inv c4inv c5inv
Akaike’s information criterion and Bayesian information criterion

<table>
<thead>
<tr>
<th>Model</th>
<th>N</th>
<th>ll(null)</th>
<th>ll(model)</th>
<th>df</th>
<th>AIC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>c2inv</td>
<td>145</td>
<td>-1702.554</td>
<td>10</td>
<td>3425.108</td>
<td>3454.876</td>
<td></td>
</tr>
<tr>
<td>c3inv</td>
<td>145</td>
<td>-1653.238</td>
<td>14</td>
<td>3334.476</td>
<td>3376.15</td>
<td></td>
</tr>
<tr>
<td>c4inv</td>
<td>145</td>
<td>-1626.828</td>
<td>18</td>
<td>3289.656</td>
<td>3343.237</td>
<td></td>
</tr>
<tr>
<td>c5inv</td>
<td>145</td>
<td>-1578.207</td>
<td>22</td>
<td>3200.414</td>
<td>3265.902</td>
<td></td>
</tr>
</tbody>
</table>
```

Note: BIC uses N = number of observations. See [R] BIC note.

The model with five latent classes has the smallest values of both AIC and BIC and would be considered the best based on these information criteria.

**Fitting the three-class model with covariances**

*Masyn’s* final model was a three-class model that allowed for covariances among the error terms and that estimated all parameters separately across classes. To estimate the covariances, we add the `covstructure(e._OEn, unstructured)` option. See [SEM] sem and gsem option covstructure() for details on this option. To allow all parameters to vary across classes, we add the `lcinvariant(none)` option. Here `none` specifies that no parameters are constrained to be equal across classes.

```
. gsem (glucose insulin sspg <- _cons), lclass(C 3) lcinvariant(none)  
>              > covstructure(e._OEn, unstructured)
(iteration log omitted)
Generalized structural equation model Number of obs = 145
Log likelihood = -1536.6409

|            | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------|--------|-----------|-------|------|----------------------|
| 1.C        | (base outcome) |        |       |      |                      |
| 2.C        | _cons  | -.8853513 | .2386536 | -3.71 | 0.000 | -1.353104 - .4175988 |
| 3.C        | _cons  | -.612664  | .2260018 | -2.71 | 0.007 | -1.055619 - .1697085 |
```
Example 52g — Latent profile model

<table>
<thead>
<tr>
<th>Class</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>glucose</td>
</tr>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
<tr>
<td>Response</td>
<td>insulin</td>
</tr>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
<tr>
<td>Response</td>
<td>sspg</td>
</tr>
<tr>
<td>Family</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Link</td>
<td>identity</td>
</tr>
</tbody>
</table>

|                | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|----------------|-------|-----------|---|-----|----------------------|
| glucose _cons  | 35.68584 | .5741752  | 62.15 | 0.000 | 34.56048 36.81121 |
| insulin _cons  | 16.58066 | .6204724  | 26.72 | 0.000 | 15.36456 17.79677 |
| sspg _cons     | 10.49755 | .5833606  | 17.99 | 0.000 | 9.354183 11.64091 |
| var(e.gluc~e)  | 19.30952 | 3.932547 | 12.9544 | 28.78233 |
| var(e.insu~n)  | 26.7354 | 4.494093 | 19.23108 | 37.16804 |
| var(e.sspg)    | 18.71079 | 3.970509 | 12.34422 | 28.36094 |
| cov(e.gluc~e, e.insulin) | 3.456027 | 2.942391 | 1.17 | 0.240 | -2.310954 9.223008 |
| cov(e.gluc~e, e.sspg) | 5.474303 | 2.811729 | 1.95 | 0.052 | -.0365846 10.98519 |
| cov(e.insu~n, e.sspg) | 7.995803 | 3.020304 | 2.65 | 0.008 | 2.076115 13.91549 |
Class: 2  
Response: glucose  
Family: Gaussian  
Link: identity  
Response: insulin  
Family: Gaussian  
Link: identity  
Response: sspg  
Family: Gaussian  
Link: identity

|                | Coef. | Std. Err. | z     | P>|z|    | [95% Conf. Interval] |
|----------------|-------|-----------|-------|--------|---------------------|
| glucose_\_cons | 47.66176 | 1.492718 | 31.93 | 0.000  | 44.73609 50.58744   |
| insulin_\_cons | 34.35203 | 3.00337  | 11.44 | 0.000  | 28.46554 40.23853  |
| sspg_\_cons    | 24.414 | .7395383  | 33.01 | 0.000  | 22.96453 25.86347  |
| var(e.gluc\_e) | 53.21326 | 15.56547 |       |        | 29.99396 94.40735  |
| var(e.insu\_n) | 228.6332 | 59.03553 |       |        | 137.832 379.2526   |
| var(e.sspg)    | 13.75515 | 3.838523 |       |        | 7.960284 23.76853  |
| cov(e.gluc\_e, e.insulin) | 40.02875 | 23.12762 | 1.73  | 0.083  | -5.300552 85.35805 |
| cov(e.gluc\_e, e.sspg) | .7294854 | 5.48065 | 0.13  | 0.894  | -10.01239 11.47136 |
| cov(e.insu\_n, e.sspg) | -5.743169 | 11.4943 | -0.50 | 0.617  | -28.27158 16.78524 |
Example 52g — Latent profile model

Class: 3
Response: glucose
Family: Gaussian
Link: identity
Response: insulin
Family: Gaussian
Link: identity
Response: sspg
Family: Gaussian
Link: identity

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|-------|---------------------|
| glucose | 93.92473 | 6.985336 | 13.45 | 0.000 | 80.23372 | 107.6157 |
| insulin | 10.37614 | 1.123135 | 9.24 | 0.000 | 8.174836 | 12.57744 |
| sspg | 28.4787 | 1.94975 | 14.61 | 0.000 | 24.65726 | 32.30013 |
| var(e.glucose) | 1279.011 | 312.6774 | 792.1048 | 2065.218 |
| var(e.insulin) | 36.38521 | 9.26287 | 22.09163 | 59.92692 |
| var(e.sspg) | 113.3239 | 27.67628 | 70.21642 | 182.8961 |
| cov(e.glucose, e.insulin) | -163.4383 | 47.637 | -3.43 | 0.001 | -256.8051 | -70.07153 |
| cov(e.glucose, e.sspg) | 276.9206 | 81.60543 | 3.39 | 0.001 | 116.9769 | 436.8643 |
| cov(e.insulin, e.sspg) | 25.4313 | 11.66564 | -2.18 | 0.029 | -48.29554 | -2.567057 |

Because we do not have any predictors in our regression models, the intercepts can be interpreted as the predicted class-specific means of the corresponding variables. In class 1, glucose has an estimated mean of 35.69, insulin has an estimated mean of 16.58, and sspg has an estimated mean of 10.50. Also because we have no predictors, the estimated variances and covariances of the error terms are simply class-specific estimates of the variances and covariances of the variables. In class 1, the estimated variance of glucose is 19.31, the estimated covariance of glucose and insulin is 3.46. The remaining coefficients can be interpreted in a similar manner.

We can determine expected classification for each individual in the dataset based on the predicted posterior class probabilities.

. predict cpost*, classposteriorpr
. egen max = rowmax(cpost*)
. generate predclass = 1 if cpost1==max
   (69 missing values generated)
. replace predclass = 2 if cpost2==max
   (32 real changes made)
. replace predclass = 3 if cpost3==max
   (37 real changes made)
### Example 52g — Latent profile model

When we compare the predicted classes (predclass) with the assigned clinical classifications (cclass) given to these individuals, we see that 91% of the individuals predicted to be in class 1 were given a clinical classification of normal. Of those predicted to be in class 2, 72% were assigned a clinical classification of chemical diabetic. Finally, 84% of those predicted to be in class 3 had a clinical classification of overt diabetic.

Masyn went on to examine the individuals who were classified differently when using the clinical definition and when using the results from the model. She found that the predictions from the latent profile model could be explained medically and may be an improvement over the clinical definitions.

### References


### Also see

- [SEM] **Example 50g** — Latent class model
- [SEM] **Example 51g** — Latent class goodness-of-fit statistics
- [SEM] **Intro 5** — Tour of models
- [SEM] **gsem** — Generalized structural equation model estimation command

<table>
<thead>
<tr>
<th>Clinical classification</th>
<th>predclass</th>
<th></th>
<th></th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>Total</td>
<td></td>
</tr>
<tr>
<td>overt diabetic</td>
<td>0</td>
<td>2</td>
<td>31</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.00</td>
<td>6.25</td>
<td>83.78</td>
<td>22.76</td>
<td></td>
</tr>
<tr>
<td>chemical diabetic</td>
<td>7</td>
<td>23</td>
<td>6</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9.21</td>
<td>71.88</td>
<td>16.22</td>
<td>24.83</td>
<td></td>
</tr>
<tr>
<td>normal</td>
<td>69</td>
<td>7</td>
<td>0</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.79</td>
<td>21.88</td>
<td>0.00</td>
<td>52.41</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>76</td>
<td>32</td>
<td>37</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td></td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td></td>
</tr>
</tbody>
</table>
Example 53g — Finite mixture Poisson regression

Description

To demonstrate a finite mixture model (FMM), we use the following data:

```
use https://www.stata-press.com/data/r16/gsem_mixture
(U.S. Medical Expenditure Panel Survey (2003))
```

. use https://www.stata-press.com/data/r16/gsem_mixture
(U.S. Medical Expenditure Panel Survey (2003))

Contains data from https://www.stata-press.com/data/r16/gsem_mixture.dta

| vars: 12   | 26 Jan 2019 08:46 |
| _dta has notes |

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display format</th>
<th>value label</th>
</tr>
</thead>
<tbody>
<tr>
<td>drvisits</td>
<td>int</td>
<td>%9.0g</td>
<td>number of doctor visits</td>
</tr>
<tr>
<td>private</td>
<td>byte</td>
<td>%8.0g</td>
<td>has private supplementary insurance</td>
</tr>
<tr>
<td>medicaid</td>
<td>byte</td>
<td>%8.0g</td>
<td>has Medicaid public insurance</td>
</tr>
<tr>
<td>age</td>
<td>byte</td>
<td>%8.0g</td>
<td>age in years</td>
</tr>
<tr>
<td>educ</td>
<td>byte</td>
<td>%8.0g</td>
<td>years of education</td>
</tr>
<tr>
<td>actlim</td>
<td>byte</td>
<td>%8.0g</td>
<td>has activity limitations</td>
</tr>
<tr>
<td>chronic</td>
<td>byte</td>
<td>%8.0g</td>
<td>number of chronic conditions</td>
</tr>
<tr>
<td>income</td>
<td>float</td>
<td>%9.0g</td>
<td>income in $1,000s</td>
</tr>
<tr>
<td>offer</td>
<td>byte</td>
<td>%8.0g</td>
<td>employer offers insurance</td>
</tr>
<tr>
<td>hpvisits</td>
<td>int</td>
<td>%8.0g</td>
<td>number of visits to health professionals other than doctors</td>
</tr>
<tr>
<td>female</td>
<td>byte</td>
<td>%8.0g</td>
<td>female</td>
</tr>
<tr>
<td>phylim</td>
<td>byte</td>
<td>%8.0g</td>
<td>has physical limitation</td>
</tr>
</tbody>
</table>

Sorted by:

. notes

_dta:

1. Data on annual number of doctor visits for individuals age 65 and older from the U.S. Medical Expenditure Panel Survey for 2003.
2. Data are analyzed in Cameron, A. C. and P. K. Trivedi, 2010, _Microeconometrics Using Stata, Rev. Ed._, College Station, TX: Stata Press.
3. Additional information on finite mixture models for count data and a similar example are found in Deb, P. and P. K. Trivedi, 1997, Demand for medical care by the elderly: A finite mixture approach, _Journal of Applied Econometrics_, vol. 12, 313--336.

See _Finite mixture models_ in [SEM] _Intro 5_ for background.
Remarks and examples

We are interested in fitting a Poisson regression to model the annual number of doctor visits as a function of whether an individual has private supplementary insurance, whether he or she has Medicaid, age, age squared, education level, whether he or she has activity limitations, and the number of chronic conditions. If we believed that the same model applied to the entire population, we could fit the model by typing

```
.poisson drvisits private medicaid c.age##c.age educ actlim chronic
```
or, equivalently, by using `gsem`

```
.gsem (drvisits <- private medicaid c.age##c.age educ actlim chronic), poisson
```

However, we believe that the model may differ across groups in the population. We do not have any information that identifies what these groups are or that tells us which individuals in our sample belong to each group. We can consider a categorical latent variable that identifies these groups and refer to the levels of this latent variable as latent classes. With an FMM, we can incorporate the categorical latent variable into our model to account for differences across the latent classes.

Following Cameron and Trivedi (2010), we will fit an FMM with a Poisson regression component for each latent class. We will estimate distinct coefficients for the Poisson model in each class, and we will estimate the probability of belonging to each of these classes using a multinomial logistic regression. We fit the model as follows:

```
.gsem (drvisits <- private medicaid c.age##c.age educ actlim chronic),
> poisson lclass(C 2) startvalues(randomid, draws(5) seed(15))
```

Computing starting values using randomid:

```
( iteration log omitted )
```

Generalized structural equation model

|                | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|----------------|-------|-----------|------|------|----------------------|
| 1.C            |       |           |      |      |                      |
| 2.C _cons      | .877227 | .0494614  | 17.74 | 0.000 | .7802845 .9741696    |

Class : 1
Response : drvisits
Family : Poisson
Link : log

|                | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|----------------|-------|-----------|------|------|----------------------|
| drvisits       |       |           |      |      |                      |
| private        | .138229 | .0247626  | 5.58 | 0.000 | .0896951 .1867629    |
| medicaid       | .1269723 | .0341525 | 3.72 | 0.000 | .0600345 .19391     |
| age            | .2628874 | .0466774  | 5.63 | 0.000 | .1714014 .3543735    |
| c.age#c.age    | -.0017418 | .0003108 | -5.60 | 0.000 | -.002351 -.0011326   |
| educ           | .0241679 | .0030705 | 7.87 | 0.000 | .0181499 .030186     |
| actlim         | .1831598 | .0238817  | 7.67 | 0.000 | .1363525 .2299671    |
| chronic        | .1970511 | .0088783  | 22.19 | 0.000 | .17965 .2144523     |
| _cons          | -8.051256 | 1.741677 | -4.62 | 0.000 | -11.46488 -4.637632  |
Example 53g — Finite mixture Poisson regression

Class : 2
Response : drvisits
Family : Poisson
Link : log

| Coef.     | Std. Err. | z      | P>|z| | [95% Conf. Interval] |
|-----------|-----------|--------|------|----------------------|
| drvisits  |           |        |      |                      |
| private   | .2077415  | .0306353 | 6.78      | .1476974 | .2677856 |
| medicaid  | .1071618  | .0407211 | 2.63    | .02735 | .1869736 |
| age       | .3798087  | .0562035 | 6.76    | .269652 | .4899655 |
| c.age#c.age | -.0024869 | .0003736 | -6.66  | .000 | -.0032191 | -.0017547 |
| educ      | .029099   | .003972 | 7.33   | .021314 | .0368841 |
| actlim    | .1244235  | .0310547 | 4.01  | .0635574 | .1852895 |
| chronic   | .3191166  | .0089757 | 35.55 | .3015247 | .3367086 |
| _cons     | -14.25713 | 2.101964 | -6.78 | .18.37691 | -10.13736 |

Notes:
1. We used the lclass(C 2) to specify that our categorical latent variable is named C and has two latent classes.
2. The first table in the output provides the estimated coefficients in the multinomial logit model for C.
3. The next two tables are the results for the Poisson regression models for the first and second classes. By default, the coefficients and intercepts vary across the classes. We can specify lcinvariant(cons) if we want intercepts to be constrained to be equal across classes, or we can specify lcinvariant(coef) if we want all coefficients constrained to be equal across classes. See [SEM] gsem lclass options for details on the lcinvariant() option.
4. We added the startvalues(randomid), draws(5) seed(15)) option to request that starting values be computed using random class assignments. In this option, draws(5) specifies that five random draws be taken and that the one with the best log likelihood after the EM iterations be selected. If you fit FMMs and other models with categorical latent variables, taking multiple draws of random starting values can help to prevent convergence at a local maximum rather than the global maximum. gsem provides a variety of options for obtaining starting values. See [SEM] Intro 12 and [SEM] gsem estimation options for more information on starting values.
5. The fmm: prefix can be used to fit finite mixture regression models with a single response variable. We could have fit this same model with fmm: poisson by typing

```bash
.fmm 2, startvalues(randomid, draws(5) seed(15)): ///
poisson drvisits private medicaid c.age##c.age educ actlim chronic
```
We can use `estat lcprob` to estimate the proportion of individuals in each class.

```plaintext
. estat lcprob
Latent class marginal probabilities

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin</td>
<td>Std. Err.</td>
<td>[95% Conf. Interval]</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>.2937527</td>
<td>.0102614</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>.7062473</td>
<td>.0102614</td>
</tr>
</tbody>
</table>
```

We find that about 29% of the population is in class 1 and about 71% is in class 2.

To better understand these classes, we use `estat lcmean` to estimate the marginal predicted counts (means) for each class.

```plaintext
. estat lcmean
Latent class marginal means

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin</td>
<td>Std. Err.</td>
<td>z</td>
</tr>
<tr>
<td>1</td>
<td>drvisits</td>
<td>13.95943</td>
<td>.1767506</td>
</tr>
<tr>
<td>2</td>
<td>drvisits</td>
<td>3.801692</td>
<td>.0587685</td>
</tr>
</tbody>
</table>
```

Class 1 appears to represent those who visit the doctor frequently and class 2, those who visit less frequently.

We can also predict the posterior probabilities of class membership and then use those to determine the predicted class for each individual.

```plaintext
. predict postpr_dr*, classposteriorpr
. generate pclass_dr = 1 + (postpr_dr2>0.5)
. tabulate pclass_dr

<table>
<thead>
<tr>
<th></th>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,061</td>
<td>28.86</td>
<td>28.86</td>
</tr>
<tr>
<td>2</td>
<td>2,616</td>
<td>71.14</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>3,677</td>
<td>100.00</td>
<td></td>
</tr>
</tbody>
</table>
```

We see that 1,061 individuals in our sample are predicted to be in class 1, the class that frequently visits the doctor.

Our dataset also includes the variable `hpvisits`, which records the number of visits individuals make to health professionals other than doctors. We fit a similar model to the one above but with `hpvisits` as our response variable.
Example 53g — Finite mixture Poisson regression

```
.gsem (hpvisits <- private medicaid c.age##c.age educ actlim chronic),
> poisson lclass(C 2) startvalues(classid pclass_dr)
(Iteration log omitted)
```

Generalized structural equation model

|                         | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------------------|-------|-----------|-----|-----|----------------------|
| 1.C (base outcome)      |       |           |     |     |                      |
| 2.C                     | 2.241837 | .059523 | 37.66 | 0.000 | 2.125174 2.3585      |

Class: 1
Response: hpvisits
Family: Poisson
Link: log

```
|                         | Coef.     | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------------------|-----------|-----------|-----|-----|----------------------|
| hpvisits                |           |           |     |     |                      |
| private                 | .3218525  | .0347116  | 9.27| 0.000| .253819 .389886      |
| medicaid                | .0715449  | .0566317  | 1.26| 0.206| -.0394511 .182541    |
| age                     | .0975749  | .0743567  | 1.31| 0.189| -.0481615 .2433113   |
| c.age##c.age            | -0.0004749 | .0004971 | -0.96| 0.339| -.0014492 .0004993   |
| educ                    | .0278151  | .0046572  | 5.97| 0.000| .0186872 .0369429    |
| actlim                  | .7088077  | .0353277  | 20.06| 0.000| .6395666 .7780488    |
| chronic                 | -0.0077779 | .0127981 | -0.61| 0.543| -.0328617 .0173059   |
| _cons                   | -2.430713 | 2.766794  | -0.88| 0.380| -7.853529 2.992103   |

Class: 2
Response: hpvisits
Family: Poisson
Link: log

```
|                         | Coef.     | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------------------|-----------|-----------|-----|-----|----------------------|
| hpvisits                |           |           |     |     |                      |
| private                 | .4448319  | .0451971  | 9.84| 0.000| .3562473 .5334165    |
| medicaid                | -.4490187 | .0742525  | -6.05| 0.000| -.5945499 -.3034875  |
| age                     | .4160345  | .0797576  | 5.22| 0.000| .2597125 .5723565    |
| c.age##c.age            | -.0026784 | .0005287  | -5.07| 0.000| -.0037147 -.0016421  |
| educ                    | .1250644  | .0062982  | 19.88 | 0.000| .1127322 .1373967    |
| actlim                  | .3357366  | .0442285  | 7.59 | 0.000| .2490503 .4224229    |
| chronic                 | .2065855  | .0152161  | 13.58| 0.000| .176762 .2364081     |
| _cons                   | -18.21906 | 2.991859  | -6.09| 0.000| -24.083 -12.35513    |

This time, we used the startvalues(classid pclass_dr) option to specify how starting values are calculated. This means that we are using the variable pclass_dr as an initial guess of class membership to be used when computing starting values.
We again use `estat lcprob` to estimate the predicted proportion of the population in each class. 

```
. estat lcprob
Latent class marginal probabilities
Number of obs = 3,677

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin</td>
<td>Std. Err.</td>
<td>[95% Conf. Interval]</td>
</tr>
<tr>
<td>C</td>
<td>.0960559</td>
<td>.0051683</td>
<td>.0863925 .106674</td>
</tr>
<tr>
<td>1</td>
<td>.9039441</td>
<td>.0051683</td>
<td>.893326 .9136075</td>
</tr>
</tbody>
</table>
```

This time about 10% is in class 1, and 90% is in class 2.

We can predict the class for each individual based on this model and compare the classifications from the two models.

```
. predict postpr_hp*, classposteriorpr
. generate pclass_hp = 1 + (postpr_hp2>0.5)
. tabulate pclass_hp pclass_dr
```

```
pclass_hp    pclass_dr
            1   2   Total
1      169  180   349
2      892 2,436 3,328
Total  1,061 2,616 3,677
```

Many individuals are predicted to be in class 2 based on both models, meaning that they are in the group that visits the doctor infrequently and in the group that visits other health professionals infrequently. However, there are also 892 that are classified differently by the two models. These individuals are in the class that visits the doctor frequently based on the first model but in the class that visits other healthcare professionals infrequently based on the second model.

In [SEM] Example 54g, we consider simultaneously modeling `drvisits` and `hpvisits` and using a single categorical latent variable that identifies groups in the population.

References


Also see

[SEM] Example 54g — Finite mixture Poisson regression, multiple responses

[SEM] Intro 5 — Tour of models

[SEM] gsem — Generalized structural equation model estimation command

[SEM] estat lcmean — Latent class marginal means

[SEM] estat lcprob — Latent class marginal probabilities

[FMM] fmm intro — Introduction to finite mixture models

[FMM] fmm: poisson — Finite mixtures of Poisson regression models
Example 54g — Finite mixture Poisson regression, multiple responses

Description

In this example, we demonstrate how to fit a finite mixture model with more than one response variable.

We continue with [SEM] Example 53g, where we fit two separate finite mixture Poisson regression models, one for drvisits and one for hpvisits. To refit these models, we could type

```
use https://www.stata-press.com/data/r16/gsem_mixture
.gsem (drvisits <- private medicaid c.age##c.age educ actlim chronic), poisson lclass(C 2) startvalues(randomid, draws(5) seed(15))
.predict postpr_dr*, classposteriorpr
.generate pclass_dr = 1 + (postpr_dr2>0.5)
.gsem (hpvisits <- private medicaid c.age##c.age educ actlim chronic), poisson lclass(C 2) startvalues(classid pclass_dr)
```

As we pointed out in [SEM] Example 53g, we could have fit these finite mixture models using `fmm: poisson`. `gsem` extends the types of models that can be fit with the `fmm: prefix`, and we demonstrate one possible extension here.

See Finite mixture models in [SEM] Intro 5 for background.

Remarks and examples

We fit Poisson regression models for drvisits and hpvisits simultaneously and include one categorical latent variable C that has three classes.

```
.gsem (drvisits hpvisits <- private medicaid c.age##c.age educ actlim chronic), poisson lclass(C 3) startvalues(randomid, draws(5) seed(15))
```

```
507
```

| Class | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-------|-----------|-------|------|----------------------|
| 1.C   |       |           |       |      | (base outcome)       |
| 2.C   | _cons | -.7734177 | .0645659 | -11.98 | 0.000 | -.8999645 -.6468708 |
| 3.C   | _cons | .7802324  | .0514513 | 15.16 | 0.000 | .6793898 .8810751  |

Class: 1
Response: drvisits
Family: Poisson
Link: log
Response: hpvisits
Family: Poisson
Link: log
### Table 1: Coefficients for Drvisits

|          | Coef.   | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|----------|---------|-----------|--------|------|----------------------|
| drvisits |         |           |        |      |                      |
| private  | 0.1474015 | 0.0300291 | 4.91   | 0.000 | 0.0885456 - 0.2062574 |
| medicaid | 0.8527038 | 0.0562373 | 15.16  | 0.000 | 0.7424807 - 0.9629269 |
| age      | 0.3724257 | 0.0592566 | 6.66   | 0.000 | 0.2628137 - 0.4820378 |
| c.age#c.age | -0.0024633 | 0.0003717 | -6.63  | 0.000 | -0.0031918 - 0.0017348 |
| educ     | 0.0245594 | 0.0407563 | 6.03   | 0.000 | 0.0165713 - 0.0325474 |
| actlim   | -1.245655 | 0.0355763 | -35.01 | 0.000 | -1.315383 - 1.175927  |
| chronic  | 0.2613438 | 0.0110305 | 23.69  | 0.000 | 0.2397244 - 0.2829633 |
| _cons    | -12.35694 | 2.091664  | -5.91  | 0.000 | -16.45652 - 8.257351  |

### Table 2: Coefficients for Hpvisits

|          | Coef.   | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|----------|---------|-----------|--------|------|----------------------|
| hpvisits |         |           |        |      |                      |
| private  | 0.9375808 | 0.0867609 | 10.81  | 0.000 | 0.7675326 - 1.107629 |
| medicaid | 3.677892  | 0.0923084 | 39.96  | 0.000 | 3.4975  - 3.858248  |
| age      | 0.711631  | 0.1290143 | 5.51   | 0.000 | 0.4582996 - 0.9640266 |
| c.age#c.age | -0.0047949 | 0.0008583 | -5.59  | 0.000 | -0.0064771 - 0.0031127 |
| educ     | 0.0198688 | 0.0079941 | 2.49   | 0.013 | 0.0042008 - 0.0355369 |
| actlim   | -0.3651267| 0.065141  | -5.61  | 0.000 | -0.4928007 - 0.2374527 |
| chronic  | 0.005002  | 0.0043097 | 1.17   | 0.247 | 0.0016985 - 0.0083033 |
| _cons    | -26.74904 | 4.815349  | -5.55  | 0.000 | -36.18695 - 17.31112 |

Class: 2
Response: drvisits
Family: Poisson
Link: log
Response: hpvisits
Family: Poisson
Link: log

### Table 3: Coefficients for Drvisits (Continued)

|          | Coef.   | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|----------|---------|-----------|--------|------|----------------------|
| drvisits |         |           |        |      |                      |
| private  | 0.0147941 | 0.0388253 | 0.38   | 0.703 | -0.0613021 - 0.0908902 |
| medicaid | 0.3523114 | 0.0436849 | 8.06   | 0.000 | 0.2666905 - 0.4379323 |
| age      | 0.0498856 | 0.0732193 | 0.68   | 0.496 | -0.0936215 - 0.1933927 |
| c.age#c.age | -0.003573 | 0.0004861 | -0.73  | 0.462 | -0.0013101 - 0.0005955 |
| educ     | 0.026515 | 0.0043097 | 6.15   | 0.000 | 0.0180681 - 0.0349618 |
| actlim   | 0.4746876 | 0.0355211 | 13.47  | 0.000 | 0.4055968 - 0.5437785 |
| chronic  | 0.1662376 | 0.0125336 | 13.26  | 0.000 | 0.1416721 - 0.1908031 |
| _cons    | -0.2725728 | 2.738699 | -0.10  | 0.919 | -5.640324 - 5.096179 |

### Table 4: Coefficients for Hpvisits (Continued)

|          | Coef.   | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|----------|---------|-----------|--------|------|----------------------|
| hpvisits |         |           |        |      |                      |
| private  | 0.2289702 | 0.0345173 | 6.63   | 0.000 | 0.1613176 - 0.2966228 |
| medicaid | -2.987888 | 0.1067435 | -27.99 | 0.000 | -3.197084 - 2.779692 |
| age      | -0.0789968 | 0.0701865 | -1.13  | 0.260 | -0.2165598 - 0.0585661 |
| c.age#c.age | 0.000674 | 0.0004644 | 1.45   | 0.147 | -0.0002363 - 0.0015842 |
| educ     | 0.0519305 | 0.0048815 | 10.64  | 0.000 | 0.0423629 - 0.0614981 |
| actlim   | 0.7552881 | 0.0355466 | 21.25  | 0.000 | 0.6855168 - 0.8249582 |
| chronic  | 0.0077406 | 0.0129495 | 0.60   | 0.550 | -0.01764 - 0.0331213 |
| _cons    | 4.051195  | 2.634801  | 1.54   | 0.124 | -1.112919 - 9.21531 |
To better understand the three classes, we can use `estat lcmean` to obtain the marginal counts for doctor visits and for visits to other health professionals in all three classes.

```
. estat lcmean
Latent class marginal means
Number of obs = 3,677
```

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th>Number of obs = 3,677</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin Std. Err.</td>
<td>z</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drvisits</td>
<td>10.53552 .1875668 56.17 0.000 10.1679 10.90314</td>
<td></td>
</tr>
<tr>
<td>hpvisits</td>
<td>5.288516 .2145266 24.65 0.000 4.868052 5.70898</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drvisits</td>
<td>10.61758 .1903746 55.77 0.000 10.24445 10.99071</td>
<td></td>
</tr>
<tr>
<td>hpvisits</td>
<td>16.8819 .3076167 54.88 0.000 16.27898 17.48482</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drvisits</td>
<td>5.665389 .0735717 77.01 0.000 5.521191 5.809587</td>
<td></td>
</tr>
<tr>
<td>hpvisits</td>
<td>1.230097 .0325677 37.77 0.000 1.166265 1.293928</td>
<td></td>
</tr>
</tbody>
</table>

The first class appears to be a group that visits the doctor frequently but visits other health professionals less frequently. The second class represents those who visit both the doctor and other health professionals frequently, and the third class represents those who visit both the doctor and other health professionals less frequently.
Finally, we use `estat lcprob` to determine the expected proportions in each class.

```
. estat lcprob
Latent class marginal probabilities
Number of obs = 3,677

|       | Delta-method
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Margin</td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>.2744679</td>
</tr>
<tr>
<td>2</td>
<td>.1266487</td>
</tr>
<tr>
<td>3</td>
<td>.5988834</td>
</tr>
</tbody>
</table>
```

Based on this model, we expect 27% of individuals to be in the first class, 13% to be in the second class, and 60% to be in the third class.

The estimation of standard errors for marginal means and marginal probabilities can be time-consuming with large models. If you are interested only in the means and probabilities, you can specify the `nose` option with `estat lcmean` and `estat lcprob` to speed up estimation. With this option, no standard errors, test statistics, or confidence intervals are reported.

### References


### Also see

- [SEM] `Example 53g` — Finite mixture Poisson regression
- [SEM] `Intro 5` — Tour of models
- [SEM] `gsem` — Generalized structural equation model estimation command
- [SEM] `estat lcmean` — Latent class marginal means
- [SEM] `estat lcprob` — Latent class marginal probabilities
- [FMM] `fmm intro` — Introduction to finite mixture models
- [FMM] `fmm: poisson` — Finite mixtures of Poisson regression models
Title

gsem — Generalized structural equation model estimation command

Description

gsem fits generalized SEMs. When you use the Builder in gsem mode, you are using the gsem command.

Menu

Statistics > SEM (structural equation modeling) > Model building and estimation

Syntax

gsem paths [ if ] [ in ] [ weight ] [ , options ]

where paths are the paths of the model in command-language path notation; see [SEM] sem and gsem path notation.

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model_description_options</td>
<td>fully define, along with paths, the model to be fit</td>
</tr>
<tr>
<td>group_options</td>
<td>fit model for different groups</td>
</tr>
<tr>
<td>lclass_options</td>
<td>fit model with latent classes</td>
</tr>
<tr>
<td>estimation_options</td>
<td>method used to obtain estimation results</td>
</tr>
<tr>
<td>reporting_options</td>
<td>reporting of estimation results</td>
</tr>
<tr>
<td>syntax_options</td>
<td>controlling interpretation of syntax</td>
</tr>
</tbody>
</table>

Factor variables and time-series operators are allowed.
bootstrap, by, jackknife, permute, statsby, and svy are allowed; see [U] 11.1.10 Prefix commands.
Weights are not allowed with the bootstrap prefix; see [R] bootstrap.
vce() and weights are not allowed with the svy prefix; see [SVY] svy.
fweights, iweights, and pweights are allowed; see [U] 11.1.6 weight.
Also see [SEM] gsem postestimation for features available after estimation.

Options

model_description_options describe the model to be fit. The model to be fit is fully specified by paths—which appear immediately after gsem—and the options covariance(), variance(), and means(). See [SEM] gsem model description options and [SEM] sem and gsem path notation.
group_options allow the specified model to be fit for different subgroups of the data, with some parameters free to vary across groups and other parameters constrained to be equal across groups. See [SEM] gsem group options.

lclass_options allow the specified model to be fit across a specified number of latent classes, with some parameters free to vary across classes and other parameters constrained to be equal across classes. See [SEM] gsem lclass options.

estimation_options control how the estimation results are obtained. These options control how the standard errors (VCE) are obtained and control technical issues such as choice of estimation method. See [SEM] gsem estimation options.

reporting_options control how the results of estimation are displayed. See [SEM] gsem reporting options.

syntax_options control how the syntax that you type is interpreted. See [SEM] sem and gsem syntax options.

Remarks and examples

gsem provides important features not provided by sem and correspondingly omits useful features provided by sem. The differences in capabilities are the following:

1. gsem allows generalized linear response functions as well as the linear response functions allowed by sem.
2. gsem allows for multilevel models, something sem does not.
3. gsem allows for categorical latent variables, which are not allowed by sem.
4. gsem allows Stata’s factor-variable notation to be used in specifying models, something sem does not.
5. gsem’s method ML is sometimes able to use more observations in the presence of missing values than can sem’s method ML. Meanwhile, gsem does not provide the MLMV method provided by sem for explicitly handling missing values.
6. gsem cannot produce standardized coefficients.
7. gsem cannot use summary statistic datasets (SSDs); sem can.

gsem has nearly identical syntax to sem. Differences in syntax arise because of differences in capabilities. The resulting differences in syntax are the following:

1. gsem adds new syntax to paths to handle latent variables associated with multilevel modeling.
2. gsem adds new options to handle the family and link of generalized linear responses.
3. gsem adds new syntax to handle categorical latent variables.
4. gsem deletes options related to features it does not have, such as SSDs.
5. gsem adds technical options for controlling features not provided by sem, such as numerical integration (quadrature choices), number of integration points, and a number of options dealing with starting values, which are a more difficult proposition in the generalized SEM framework.

For a readable explanation of what gsem can do and how to use it, see the intro sections. You might start with [SEM] Intro 1.

For examples of gsem in action, see the example sections. You might start with [SEM] Example 1.

For detailed syntax and descriptions, see the references below.
Remarks on three advanced topics are presented under the following headings:

Default normalization constraints
Default covariance assumptions
How to solve convergence problems

Default normalization constraints

`gsem` applies the same rules as `sem` to identify models; see [SEM] `sem` and see [SEM] Intro 4. Everything said there about continuous latent variables applies to multilevel latent variables such as `M1[school]` and `M2[school>teacher]`.

Default covariance assumptions

`gsem` assumes the same covariance structures as does `sem`; see [SEM] `sem` and see [SEM] Intro 4. `gsem`, however, treats covariances between observed exogenous variables as given. Actually, so does `sem` unless you specify an override. The override cannot be specified with `gsem`.

How to solve convergence problems

See [SEM] Intro 12.

Stored results

`gsem` stores the following in `e()`:

Scalars
- `e(N)` number of observations
- `e(N_clust)` number of clusters
- `e(N_groups)` number of groups
- `e(k)` number of parameters
- `e(k_cat#)` number of categories for the #th depvar, ordinal
- `e(k_dv)` number of dependent variables
- `e(k_eq)` number of equations in `e(b)`
- `e(k_out#)` number of outcomes for the #th depvar, mlogit
- `e(k_rc)` number of covariances
- `e(k_rs)` number of variances
- `e(ll)` log likelihood
- `e(n_quad)` number of integration points
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if target model converged, 0 otherwise
Macros

- `e(cmd)`
- `e(cmdline)`
- `e(depvar)`
- `e(eqnames)`
- `e(wtype)`
- `e(wexp)`
- `e(fweight)`
- `e(pweight)`
- `e(iweight)`
- `e(title)`
- `e(clustvar)`
- `e(family)`
- `e(link)`
- `e(offset)`
- `e(intmethod)`
- `e(vce)`
- `e(vcetype)`
- `e(opt)`
- `e(which)`
- `e(method)`
- `e(ml_method)`
- `e(user)`
- `e(technique)`
- `e(datasignature)`
- `e(datasignaturevars)`
- `e(properties)`
- `e(estat_cmd)`
- `e(predict)`
- `e(covariates)`
- `e(footnote)`
- `e(groupvar)`
- `e(lclass)`
- `e(asbalanced)`
- `e(asobserved)`
- `e(marginsnotok)`
- `e(marginswtype)`
- `e(marginswexp)`
- `e(marginsdefault)`

Matrices

- `e(N)`
- `e(b)`
- `e(b_pclass)`
- `e(cat#)`
- `e(out#)`
- `e(Cns)`
- `e(ilog)`
- `e(gradient)`
- `e(V)`
- `e(V_modelbased)`
- `e(nobs)`
- `e(groupvalue)`
- `e(lclass_k_levels)`
- `e(lclass_bases)`

Functions

- `e(sample)`
References


Also see

[SEM] **Intro 1** — Introduction

[SEM] sem and gsem path notation — Command syntax for path diagrams

[SEM] gsem path notation extensions — Command syntax for path diagrams

[SEM] gsem model description options — Model description options

[SEM] gsem group options — Fitting models on different groups

[SEM] gsem lclass options — Fitting models with latent classes

[SEM] gsem estimation options — Options affecting estimation

[SEM] gsem reporting options — Options affecting reporting of results

[SEM] sem and gsem syntax options — Options affecting interpretation of syntax

[SEM] gsem postestimation — Postestimation tools for gsem

[SEM] Methods and formulas for gsem — Methods and formulas for gsem

[SVY] svy estimation — Estimation commands for survey data

[U] 20 Estimation and postestimation commands
**gsem estimation options — Options affecting estimation**

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
<th>Options</th>
<th>Remarks and examples</th>
<th>Also see</th>
</tr>
</thead>
</table>

**Description**

These options control how results are obtained, from starting values, to numerical integration (also known as quadrature), to how variance estimates are obtained.

**Syntax**

```
gsem paths ..., ... estimation_options
```

<table>
<thead>
<tr>
<th><strong>estimation_options</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>method(ml)</code></td>
<td>method used to obtain the estimated parameters; only one method available with <code>gsem</code></td>
</tr>
<tr>
<td><code>vce(vcetype)</code></td>
<td><code>vcetype</code> may be oim, opg, robust, or cluster clustvar</td>
</tr>
<tr>
<td><code>pweights(varlist)</code></td>
<td>sampling weight variables for each level</td>
</tr>
<tr>
<td><code>fweights(varlist)</code></td>
<td>frequency weight variables for each level</td>
</tr>
<tr>
<td><code>iweights(varlist)</code></td>
<td>importance weight variables for each level</td>
</tr>
<tr>
<td><code>from(matname)</code></td>
<td>specify starting values</td>
</tr>
<tr>
<td><code>startvalues(svmethod)</code></td>
<td>method for obtaining starting values</td>
</tr>
<tr>
<td><code>startgrid[ gridspec ]</code></td>
<td>perform a grid search to improve starting values</td>
</tr>
<tr>
<td><code>emopts(maxopts)</code></td>
<td>control EM algorithm for improved starting values</td>
</tr>
<tr>
<td><code>noestimate</code></td>
<td>do not fit the model; show starting values instead</td>
</tr>
<tr>
<td><code>intmethod(intmethod)</code></td>
<td>integration method</td>
</tr>
<tr>
<td><code>intpoints(#)</code></td>
<td>set the number of integration (quadrature) points</td>
</tr>
<tr>
<td><code>adaptopts(adaptopts)</code></td>
<td>options for adaptive quadrature</td>
</tr>
<tr>
<td><code>listwise</code></td>
<td>apply <code>sem</code>'s (not <code>gsem</code>'s) rules for omitting observations with missing values</td>
</tr>
<tr>
<td><code>dnumerical</code></td>
<td>use numerical derivative techniques</td>
</tr>
<tr>
<td><code>etolerance(#)</code></td>
<td>set the rescaling tolerance to # to prevent numerical overflow in models with continuous latent variables; rarely used</td>
</tr>
<tr>
<td><code>maximize_options</code></td>
<td>control maximization process for specified model; seldom used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>intmethod</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mvaghermite</code></td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default</td>
</tr>
<tr>
<td><code>mcaghermite</code></td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td><code>ghermite</code></td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td><code>laplace</code></td>
<td>Laplacian approximation</td>
</tr>
</tbody>
</table>
### adaptopts

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>no log</td>
<td>whether to display the iteration log for each numerical integral calculation</td>
</tr>
<tr>
<td>iterate(#)</td>
<td>set the maximum number of iterations of the adaptive technique; default is iterate(1001)</td>
</tr>
<tr>
<td>tolerance(#)</td>
<td>set tolerance for determining convergence of the adaptive parameters; default is tolerance(1e-8)</td>
</tr>
</tbody>
</table>

### Options

- **method(ml)** is the default and is the only method available with `gsem`. This option is included for compatibility with `sem`, which provides several methods; see [SEM] `sem` option method().

- **vce(vcetype)** specifies the technique used to obtain the variance–covariance matrix of the estimates. See [SEM] `sem` option method().

- **pweights(varlist)**, **fweights(varlist)**, and **iweights(varlist)** specify weight variables to be applied from the observation (first) level to the highest level groups. Only one of these options may be specified, and none of them are allowed with crossed models or intmethod(laplace).

  - **pweights(varlist)** specifies sampling weights and implies vce(robust) if the vce() option was not also specified.
  
  - **fweights(varlist)** specifies frequency weights.
  
  - **iweights(varlist)** specifies importance weights.

- **from(matname)**, **startvalues(svmethod)**, **startgrid[(gridspec)]**, and **emopts(maxopts)** specify overriding starting values, specify how other starting values are to be calculated, and provide the ability to improve the starting values. All of this is discussed in [SEM] Intro 12. Below we provide a technical description.

  - **from(matname)** allows you to specify starting values. See [SEM] Intro 12 and see [SEM] `sem` and `gsem` option from(). We show the syntax as from(matname), but from() has another, less useful syntax, too. An alternative to from() is init() used in the path specifications; see [SEM] `sem` and `gsem` path notation.

  - **startvalues()** specifies how starting values are to be computed. Starting values specified in from() override the computed starting values, and starting values specified via init() override both.

  Starting values options for models without categorical latent variables are as follows:

    - **startvalues(zero)** specifies that starting values are to be set to 0.
    - **startvalues(constantonly)** builds on startvalues(zero) by fitting a constant-only model for each response to obtain estimates of intercept and scale parameters, and it substitutes 1 for the variances of latent variables.
    - **startvalues(fixedonly [, maxopts])** builds on startvalues(constantonly) by fitting a full fixed-effects model for each response variable to obtain estimates of coefficients along with intercept and scale parameters, and it continues to use 1 for the variances of latent variables.
    - **startvalues(ivloadings [, maxopts])** builds on startvalues(fixedonly) by using instrumental-variable methods with the generalized residuals from the fixed-effects models to compute starting values for latent variable loadings, and still uses 1 for the variances of latent variables.
startvalues(iv[, maxopts]) builds on startvalues(ivloadings) by using instrumental-variable methods with generalized residuals to obtain variances of latent variables.

Starting values options for models with categorical latent variables are as follows:

- **startvalues(factor[, maxopts])** specifies that starting values be computed by assigning each observation to an initial latent class that is determined by running a factor analysis on all the observed variables in the specified model. This is the default for models with categorical latent variables.

- **startvalues(classid varname[, maxopts])** specifies that starting values be computed by assigning each observation to an initial latent class specified in varname. varname is required to have each class represented in the estimation sample.

- **startvalues(classpr varlist[, maxopts])** specifies that starting values be computed using the initial class probabilities specified in varlist. varlist is required to contain $k$ variables for a model with $k$ latent classes. The values in varlist are normalized to sum to 1 within each observation.

- **startvalues(randomid[, draws(#) seed(#) maxopts])** specifies that starting values be computed by randomly assigning observations to initial classes.

- **startvalues(randompr[, draws(#) seed(#) maxopts])** specifies that starting values be computed by randomly assigning initial class probabilities.

- **startvalues(jitter[#c[#v], draws(#) seed(#) maxopts])** specifies that starting values be constructed by randomly perturbing the values from a Gaussian approximation to each outcome.

  - $#c$ is the magnitude for randomly perturbing coefficients, intercepts, cutpoints, and scale parameters; the default value is 1.

  - $#v$ is the magnitude for randomly perturbing variances for Gaussian outcomes; the default value is 1.

Some starting values options have suboptions that allow for tuning the starting values calculations:

- **maxopts** is a subset of the standard maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#); see [R] Maximize.

- **draws(#)** specifies the number of random draws. For startvalues(randomid), startvalues(randompr), and startvalues(jitter), gsem will generate # random draws and select the starting values from the draw with the best log-likelihood value from the EM iterations. The default is draws(1).

- **seed(#)** sets the random-number seed.

- **startgrid()** performs a grid search on variance components of latent variables to improve starting values. This is well discussed in [SEM] Intro 12. No grid search is performed by default unless the starting values are found to be not feasible, in which case gsem runs startgrid() to perform a “minimal” search involving $3^{L}$ likelihood evaluations, where $L$ is the number of latent variables. Sometimes this resolves the problem. Usually, however, there is no problem and startgrid() is not run by default. There can be benefits from running startgrid() to get better starting values even when starting values are feasible.

- **emopts(maxopts)** controls maximization of the log likelihood for the EM algorithm. The EM algorithm is used only for models with categorical latent variables. maxopts is the same subset of maximize_options that are allowed in the startvalues() option, but some of the defaults are
different for the EM algorithm. The default maximum number of iterations is \texttt{iterate(20)}. The default coefficient vector tolerance is \texttt{tolerance(1e-4)}. The default log-likelihood tolerance is \texttt{ltolerance(1e-6)}.

\texttt{noestimate} specifies that the model is not to be fit. Instead, starting values are to be shown (as modified by the above options if modifications were made), and they are to be shown using the \texttt{coeflegend} style of output. An important use of this option is before you have modified starting values at all; you can type the following:

\begin{verbatim}
  . gsem ..., ... noestimate
  . matrix b = e(b)
  . ... (modify elements of b) ...
  . gsem ..., ... from(b)
\end{verbatim}

\texttt{intmethod(intmethod)}, \texttt{intpoints(#)}, and \texttt{adaptopts(adaptopts)} affect how integration for the latent variables is numerically calculated.

\texttt{intmethod(intmethod)} specifies the method and defaults to \texttt{intmethod(mvaghermite)}. We recommend this method, although sometimes the more computationally intensive \texttt{intmethod(mcaghermite)} works better for multilevel models that are failing to converge. Sometimes it is useful to fall back on the less computationally intensive and less accurate \texttt{intmethod(ghermite)} and \texttt{intmethod(laplace)} to get the model to converge and then perhaps use one of the other more accurate methods. All of this is explained in \cite{SEM Intro 12}. \texttt{intmethod(laplace)} is the default when fitting crossed models. Crossed models are often difficult.

\texttt{intpoints(#)} specifies the number of integration points to use and defaults to \texttt{intpoints(7)}. Increasing the number increases accuracy but also increases computational time. Computational time is roughly proportional to the number specified. See \cite{SEM Intro 12}.

\texttt{adaptopts(adaptopts)} affects the adaptive part of adaptive quadrature (another term for numerical integration) and thus is relevant only for \texttt{intmethod(mvaghermite)}, \texttt{intmethod(mcaghermite)}, and \texttt{intmethod(laplace)}.

\texttt{adaptopts()} defaults to \texttt{adaptopts(nolog iterate(1001) tolerance(1e-8))}.

\texttt{[no]log} specifies whether iteration logs are shown each time a numerical integral is calculated.

\texttt{iterate()} specifies the maximum number of iterations of the adaptive technique.

\texttt{tolerance()} specifies the tolerance for determining convergence of the adaptive parameters. Convergence is declared when the relative change in the log likelihood is less than or equal to the tolerance.

\texttt{listwise} applies \texttt{sem}'s rules rather than \texttt{gsem}'s rules for omitting observations with missing values.

By default, \texttt{gsem} is sometimes able to use observations containing missing values for fitting parts of the model. \texttt{sem}, meanwhile, applies a listwise-deletion rule unless it is using \texttt{method(mlmv)}. Specifying \texttt{listwise} allows us at StataCorp to verify that \texttt{gsem} and \texttt{sem} produce the same results. We find that reassuring. Actually, automated tests verify that results are the same before shipping. For your information, \texttt{sem} and \texttt{gsem} use different numerical machinery for obtaining results, and thus the near equality of results is a strong test that each is coded correctly. You may find \texttt{listwise} useful if you are reproducing results from another package that uses listwise deletion.

\texttt{dnumerical} specifies that during optimization, the gradient vector and Hessian matrix be computed using numerical techniques instead of analytical formulas. By default, \texttt{gsem} uses analytical formulas for computing the gradient and Hessian for all integration methods except \texttt{intmethod(laplace)}.
etolerance(#) specifies a positive tolerance used to determine when to rescale log-likelihood values that are too big to exponentiate. The formula for the default tolerance is
\[ \max(2, nLvars) \times \log(p) \]
where \( nLvars \) is the number of latent variables in the model and \( p \) is the number of quadrature points.
The rule gsem uses to determine when to rescale log-likelihood values at a given level is the following: rescale log-likelihood values outside the range
\[ [-M, M - #] \]
where \( M \) is a limit that represents values beyond which exponentiation can yield a missing value.
This option is rarely ever necessary but can be helpful for multilevel models with large group sizes.

maximize_options specify the standard and rarely specified options for controlling the maximization process; see [R] Maximize. The relevant options for gsem are difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, tolerance(#), ltolerance(#), nrtolerance(#) and nonrtolerance.

Remarks and examples

For more information on vce(), see [SEM] Intro 8 and [SEM] Intro 9.

For more information on the other options, see [SEM] Intro 12.

Also see

[SEM] gsem — Generalized structural equation model estimation command
[SEM] Intro 8 — Robust and clustered standard errors
[SEM] Intro 9 — Standard errors, the full story
[SEM] Intro 12 — Convergence problems and how to solve them
**Description**

`gsem` not only allows models of the form $y_i = x_i \beta + u_i$ but also allows

$$g\{E(y_i)\} = x_i \beta$$

$y_i \sim F$

where you can choose $F$ and $g(\cdot)$ from a menu. $F$ is called the family, and $g(\cdot)$ is called the link. One set of choices is the Gaussian distribution for $F$ and the identity function for $g(\cdot)$. In that case, `gsem` reproduces linear regression. Other combinations of $g(\cdot)$ and $F$ produce other popular models, including logit (also known as logistic regression), probit, multinomial logit, Poisson regression, and more.

**Syntax**

```plaintext
.gsem paths ... , ... family_and_link_options
```

<table>
<thead>
<tr>
<th>family_and_link_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>family(family)</code></td>
<td>distribution family; default is <code>family(gaussian)</code></td>
</tr>
<tr>
<td><code>link(link)</code></td>
<td>link function; default varies per family</td>
</tr>
<tr>
<td><code>cloglog</code></td>
<td>synonym for <code>family(bernoulli) link(cloglog)</code></td>
</tr>
<tr>
<td><code>exponential</code></td>
<td>synonym for <code>family(exponential) link(log)</code></td>
</tr>
<tr>
<td><code>gamma</code></td>
<td>synonym for <code>family(gamma) link(log)</code></td>
</tr>
<tr>
<td><code>logit</code></td>
<td>synonym for <code>family(bernoulli) link(logit)</code></td>
</tr>
<tr>
<td><code>loglogistic</code></td>
<td>synonym for <code>family(loglogistic) link(log)</code></td>
</tr>
<tr>
<td><code>lognormal</code></td>
<td>synonym for <code>family(lognormal) link(log)</code></td>
</tr>
<tr>
<td><code>llogistic</code></td>
<td>synonym for <code>family(llogistic) link(log)</code></td>
</tr>
<tr>
<td><code>lnormal</code></td>
<td>synonym for <code>family(lnormal) link(log)</code></td>
</tr>
<tr>
<td><code>mlogit</code></td>
<td>synonym for <code>family(multinomial) link(logit)</code></td>
</tr>
<tr>
<td><code>nbreg</code></td>
<td>synonym for <code>family(nbreg mean) link(log)</code></td>
</tr>
<tr>
<td><code>ocloglog</code></td>
<td>synonym for <code>family(ordinal) link(cloglog)</code></td>
</tr>
<tr>
<td><code>ologit</code></td>
<td>synonym for <code>family(ordinal) link(logit)</code></td>
</tr>
<tr>
<td><code>oprobit</code></td>
<td>synonym for <code>family(ordinal) link(probit)</code></td>
</tr>
<tr>
<td><code>poisson</code></td>
<td>synonym for <code>family(poisson) link(log)</code></td>
</tr>
<tr>
<td><code>probit</code></td>
<td>synonym for <code>family(bernoulli) link(probit)</code></td>
</tr>
<tr>
<td><code>regress</code></td>
<td>synonym for <code>family(gaussian) link(identity)</code></td>
</tr>
<tr>
<td><code>weibull</code></td>
<td>synonym for <code>family(weibull) link(log)</code></td>
</tr>
<tr>
<td><code>exposure(varname_e)</code></td>
<td>include $\ln(varname_e)$ with coefficient constrained to 1</td>
</tr>
<tr>
<td><code>offset(varname_o)</code></td>
<td>include $varname_o$ with coefficient constrained to 1</td>
</tr>
</tbody>
</table>
### Family Options

<table>
<thead>
<tr>
<th>Family</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gaussian [, options]</code></td>
<td>Gaussian (normal); the default</td>
</tr>
<tr>
<td><code>bernoulli</code></td>
<td>Bernoulli</td>
</tr>
<tr>
<td><code>beta</code></td>
<td>Beta</td>
</tr>
<tr>
<td><code>binomial [# varname]</code></td>
<td>Binomial; default number of binomial trials is 1</td>
</tr>
<tr>
<td><code>ordinal</code></td>
<td>Ordinal</td>
</tr>
<tr>
<td><code>multinomial</code></td>
<td>Multinomial</td>
</tr>
<tr>
<td><code>poisson [, poisson]</code></td>
<td>Poisson</td>
</tr>
<tr>
<td><code>nbinomial [mean constant]</code></td>
<td>Negative binomial; default dispersion is mean</td>
</tr>
<tr>
<td><code>exponential [, survival]</code></td>
<td>Exponential</td>
</tr>
<tr>
<td><code>gamma [, survival]</code></td>
<td>Gamma</td>
</tr>
<tr>
<td><code>loglogistic [, survival]</code></td>
<td>Loglogistic</td>
</tr>
<tr>
<td><code>lognormal [, survival]</code></td>
<td>Lognormal</td>
</tr>
<tr>
<td><code>weibull [, survival]</code></td>
<td>Weibull</td>
</tr>
<tr>
<td><code>pointmass #</code></td>
<td>Point-mass density at #</td>
</tr>
</tbody>
</table>

### Link Options

<table>
<thead>
<tr>
<th>Link</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>identity</code></td>
<td>Identity</td>
</tr>
<tr>
<td><code>log</code></td>
<td>Log</td>
</tr>
<tr>
<td><code>logit</code></td>
<td>Logit</td>
</tr>
<tr>
<td><code>probit</code></td>
<td>Probit</td>
</tr>
<tr>
<td><code>cloglog</code></td>
<td>Complementary log-log</td>
</tr>
</tbody>
</table>

### Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ldepvar(varname)</code></td>
<td>Lower depvar for interval-response data</td>
</tr>
<tr>
<td><code>udepvar(varname)</code></td>
<td>Upper depvar for interval-response data</td>
</tr>
<tr>
<td><code>lcensored(varname #)</code></td>
<td>Lower limit for left-censoring</td>
</tr>
<tr>
<td><code>rcensored(varname #)</code></td>
<td>Upper limit for right-censoring</td>
</tr>
</tbody>
</table>

Only allowed with `family(gaussian)` with `link(identity)`.  

### Poisson Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ltruncated(varname #)</code></td>
<td>Lower limit for left-truncation</td>
</tr>
</tbody>
</table>

### Survival Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ltruncated(varname #)</code></td>
<td>Lower limit for left-truncation</td>
</tr>
<tr>
<td><code>failure(varname)</code></td>
<td>Indicator for failure event</td>
</tr>
<tr>
<td><code>ph</code></td>
<td>Proportional hazards parameterization</td>
</tr>
<tr>
<td><code>aft</code></td>
<td>Accelerated failure-time parameterization</td>
</tr>
</tbody>
</table>

`ph` is allowed only with families `exponential` and `weibull`.  
`aft` is allowed only with families `exponential`, `gamma`, `loglogistic`, `lognormal`, and `weibull`.  

---

**Note:** The options `ldepvar`, `udepvar`, `lcensored`, and `rcensored` are only allowed with the `family(gaussian)` and `link(identity)` combination. The `ltruncated` option is only allowed with `family(poisson)`. The `failure` option is used to indicate a failure event. The `ph` and `aft` options are used for proportional hazards and accelerated failure-time parameterizations, respectively.
If you specify both family() and link(), not all combinations make sense. You may choose from the following combinations:

<table>
<thead>
<tr>
<th></th>
<th>identity</th>
<th>log</th>
<th>logit</th>
<th>probit</th>
<th>cloglog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>D</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bernoulli</td>
<td>D</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>D</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>binomial</td>
<td>D</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ordinal</td>
<td>D</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>multinomial</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>negative binomial</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>exponential</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weibull</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gamma</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>loglogistic</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lognormal</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pointmass</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D denotes the default.

Options

family(family) and link(linkname) specify $F$ and $g(\cdot)$. If neither is specified, linear regression is assumed.

Two of the families allow optional arguments:

family(binomial [ | varname ]) specifies that data are in binomial form, that is, that the response variable records the number of successes from a series of Bernoulli trials. The number of trials is given either as a constant number or as a varname that allows the number of trials to vary over observations, or it is not given at all. In the last case, the number of trials is thus equivalent to specifying family(bernoulli).

family(nbinomial [ mean | constant ]) specifies a negative binomial model, a Poisson model with overdispersion. Be aware, however, that even Poisson models can have overdispersion if latent variables are included in the model. Let’s use the term “conditional overdispersion” to refer to dispersion above and beyond that implied by latent variables, if any.

That conditional overdispersion can take one of two forms. In mean overdispersion, the conditional overdispersion is a linear function of the conditional (predicted) mean. Constant overdispersion refers to the conditional overdispersion being, of course, constant.

If you do not specify mean or constant, then mean is assumed.

family(pointmass #) is a special family allowed when lclass() is also specified.

cloglog, exponential, gamma, logit, loglogistic, lognormal, llogistic, lnormal, mlogit, nbreg, oclloglog, ologit, oprobit, poisson, probit, regress, and weibull are shorthands for specifying popular models.

exposure(varname_e) and offset(varname_o) are most commonly used with families poisson and nbreg, that is, they typically concern count models.
exposure() specifies a variable that reflects the amount of exposure—usually measured in time units—for each observation over which the responses were counted. If one observation was exposed for twice the time of another, and the observations were otherwise identical, one would expect twice as many events to be counted. To assume that, $\ln(varname)$ is entered into $x_i\beta$ with coefficient constrained to be 1.

offset() enters $varname$ into $x_i\beta$ with coefficient constrained to be 1. offset() is just another way of specifying exposure() where the offset variable is the log of amount of exposure.

If neither exposure() nor offset() is specified, observations are assumed to have equal amounts of exposure.

ldepvar(varname) and udepvar(varname) specify that each observation can be point data, interval data, left-censored data, or right-censored data. The type of data for a given observation is determined by the values in $y_i$ and varname. The following specifications are equivalent:

$$
\begin{align*}
\text{depvar}_1 & \leftarrow \ldots, \text{family}(\text{gauss, udepvar}(\text{depvar}_2)) \\
\text{depvar}_2 & \leftarrow \ldots, \text{family}(\text{gauss, ldepvar}(\text{depvar}_1))
\end{align*}
$$

Thus only one of ldepvar() or udepvar() is allowed. In either case, $\text{depvar}_1$ and $\text{depvar}_2$ should have the following form:

<table>
<thead>
<tr>
<th>Type of data</th>
<th>var1</th>
<th>var2</th>
</tr>
</thead>
<tbody>
<tr>
<td>point data</td>
<td>$a = [a, a]$</td>
<td>$a$</td>
</tr>
<tr>
<td>interval data</td>
<td>$[a, b]$</td>
<td>$a$</td>
</tr>
<tr>
<td>left-censored data</td>
<td>$(-\infty, b]$</td>
<td>.</td>
</tr>
<tr>
<td>right-censored data</td>
<td>$[a, +\infty)$</td>
<td>$a$</td>
</tr>
</tbody>
</table>

lcensored(varname|#) and rcensored(varname|#) indicate the lower and upper limits for censoring, respectively. You may specify only one.

lcensored(arg) specifies that observations with $y_i \leq arg$ are left-censored and the remaining observations are not.

rcensored(arg) specifies that observations with $y_i \geq arg$ are right-censored and the remaining observations are not.

Neither lcensored() nor rcensored() may not be combined with ldepvar() or udepvar().

ltruncated(varname | #) indicates the lower limits for truncation.

ltruncated(arg) specifies that the distribution is truncated on the left at arg, meaning that $y_i \leq arg$ is not within the support for the corresponding distribution family. This option rescales the underlying density function to accommodate the truncated support for $y_i$. This means that values of $y_i$ that are less than or equal to arg do not contribute to the likelihood. For survival families, this means that time (time at risk) starts at arg instead of at 0.
failure(varname) specifies the failure event.

If failure() is not specified, all observations are assumed to indicate a failure.

If failure(varname) is specified, varname is interpreted as an indicator variable; 0 and missing mean censored, and all other values are interpreted as representing failure.

ph specifies the proportional hazards parameterization and is allowed with families exponential and weibull. This is the default parameterization for these families.

aft specifies the accelerated failure-time parameterization and is allowed with families exponential, gamma, loglogistic, lognormal, and weibull. This is an optional parameterization for exponential and weibull but the only parameterization for the others.

Remarks and examples

In the command language, the family-and-link options may be specified at the end of a command among the global options:

. gsem ..., ... family(...) link(...) ...
. gsem ..., ... poisson exposure(time) ...

Specified that way, the options apply to all the response variables. Alternatively, they may be specified inside paths to affect single equations:

. gsem (y1 <- x1 x2, family(...) poisson(...)) (y2 <- x2 L) ...
. gsem (y1 <- x1 x2, family(...) link(...))                   ///
    (y2 <- x2 L, family(...) link(...)) ...
. gsem (y1 <- x1 x2, poisson exposure(time)) (y2 <- x2 L) ...
. gsem (y1 <- x1 x2, poisson exposure(time))                   ///
    (y2 <- x2 L, family(...) link(...)) ...

On a different topic, it is worth noting that you can fit exponential-regression models with family(gamma) link(log) if you constrain the log of the scale parameter to be 0 with gsem’s constraints() option. For instance, you might type

. constraint 1 _b[/y:logs] = 0
. gsem (y <- x1 x2, gamma), constraints(1)

The name _b[/y:logs] changes according to the name of the dependent variable. Had y instead been named waitingtime, the parameter would have been named _b[/waitingtime:logs]. Rather than remembering that, remember instead that the best way to discover the names of parameters is to type

. gsem (waitingtime <- x1 x2, gamma), noestimate

and then look at the output to discover the names. See [SEM] sem and gsem option constraints().
For examples of generalized response variables, see the following:

- **Example 27g**: Single-factor measurement model (generalized response)
- **Example 28g**: One-parameter logistic IRT (Rasch) model
- **Example 29g**: Two-parameter logistic IRT model
- **Example 30g**: Two-level measurement model (multilevel, generalized response)
- **Example 31g**: Two-factor measurement model (generalized response)
- **Example 32g**: Full structural equation model (generalized response)
- **Example 33g**: Logistic regression
- **Example 34g**: Combined models (generalized responses)
- **Example 35g**: Ordered probit and ordered logit
- **Example 36g**: MIMIC model (generalized response)
- **Example 37g**: Multinomial logistic regression
- **Example 39g**: Three-level model (multilevel, generalized response)
- **Example 41g**: Two-level multinomial logistic regression (multilevel)
- **Example 43g**: Tobit regression
- **Example 44g**: Interval regression
- **Example 45g**: Heckman selection model
- **Example 46g**: Endogenous treatment-effects model
- **Example 47g**: Exponential survival model
- **Example 48g**: Loglogistic survival model with censored and truncated data
- **Example 49g**: Multiple-group Weibull survival model
- **Example 50g**: Latent class model
- **Example 52g**: Latent profile model

Also see

- **gsem** — Generalized structural equation model estimation command
- **Intro 2** — Learning the language: Path diagrams and command language
- **sem and gsem path notation** — Command syntax for path diagrams
- **gsem path notation extensions** — Command syntax for path diagrams
**Title**

| gsem group options — Fitting models on different groups |

| Description | Syntax | Options | Remarks and examples | Also see |

**Description**

*gsem* can fit combined models across subgroups of the data while allowing some parameters to vary and constraining others to be equal across subgroups. These subgroups could be males and females, age category, and the like.

*gsem* performs such estimation when the *group(varname)* option is specified. The *ginvariant(pclassname)* option specifies which parameters are to be constrained to be equal across the groups.

**Syntax**

```
gsem paths ..., ... group_options
```

<table>
<thead>
<tr>
<th>group_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group(varname)</td>
<td>fit model for different groups</td>
</tr>
<tr>
<td>ginvariant(pclassname)</td>
<td>specify parameters that are equal across groups</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pclassname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cons</td>
<td>intercepts and cutpoints</td>
</tr>
<tr>
<td>coef</td>
<td>fixed coefficients</td>
</tr>
<tr>
<td>loading</td>
<td>latent variable coefficients</td>
</tr>
<tr>
<td>errvar</td>
<td>covariances of errors</td>
</tr>
<tr>
<td>scale</td>
<td>scaling parameters</td>
</tr>
<tr>
<td>means</td>
<td>means of exogenous variables</td>
</tr>
<tr>
<td>covex</td>
<td>covariances of exogenous latent variables</td>
</tr>
<tr>
<td>all</td>
<td>all the above</td>
</tr>
<tr>
<td>none</td>
<td>none of the above</td>
</tr>
</tbody>
</table>

*ginvariant(cons coef loading)* is the default if *ginvariant()* is not specified.

**Options**

- **group(varname)** specifies that the model be fit as described above. *varname* specifies the name of a numeric variable that records the group to which the observation belongs.

- **ginvariant(pclassname)** specifies which classes of parameters of the model are to be constrained to be equal across groups. The classes are defined above. The default is *ginvariant(cons coef loading)* if the option is not specified.
Remarks and examples

See [SEM] Intro 6 and [SEM] Example 49g.

Also see

[SEM] gsem — Generalized structural equation model estimation command
[SEM] Intro 6 — Comparing groups
[SEM] Example 49g — Multiple-group Weibull survival model
**Description**

`gsem` can fit models with categorical latent variables having specified numbers of latent classes. Some parameters can vary across classes while others are constrained to be equal across classes.

`gsem` performs such estimation when the `lclass()` option is specified. The `lcinvariant(pclassname)` option specifies which parameters are to be constrained to be equal across the latent classes.

**Syntax**

```
gsem paths ..., ... lclass(lcname # [, base(#)]) lcinvariant(pclassname)
```

<table>
<thead>
<tr>
<th><code>lclass_options</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>lclass()</code></td>
<td>fit model with latent classes</td>
</tr>
<tr>
<td><code>lcinvariant(pclassname)</code></td>
<td>specify parameters that are equal across latent classes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><code>pclassname</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cons</code></td>
<td>intercepts and cutpoints</td>
</tr>
<tr>
<td><code>coef</code></td>
<td>fixed coefficients</td>
</tr>
<tr>
<td><code>errvar</code></td>
<td>covariances of errors</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>scaling parameters</td>
</tr>
<tr>
<td><code>all</code></td>
<td>all the above</td>
</tr>
<tr>
<td><code>none</code></td>
<td>none of the above</td>
</tr>
</tbody>
</table>

`lcinvariant(errvar scale)` is the default if `lcinvariant()` is not specified.

**Options**

`lclass(lcname # [, base(#)])` specifies that the model be fit as described above.

`lcname` specifies the name of a categorical latent variable, and `#` specifies the number of latent classes. The latent classes are the contiguous integers starting with 1 and ending with `#`.

`base(#)` specifies the class of `lcname` to be treated as the base class. The default is `base(1)`.

`lcinvariant(pclassname)` specifies which classes of parameters of the model are to be constrained to be equal across the latent classes. The classes are defined above. The default is `lcinvariant(errvar scale)`. 

529
Remarks and examples

See [SEM] Intro 2, and see [SEM] Example 50g, [SEM] Example 51g, and [SEM] Example 52g.

Also see

[SEM] gsem — Generalized structural equation model estimation command
[SEM] Intro 2 — Learning the language: Path diagrams and command language
[SEM] Example 50g — Latent class model
[SEM] Example 51g — Latent class goodness-of-fit statistics
[SEM] Example 52g — Latent profile model
Title

**gsesm model description options — Model description options**

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
<th>Options</th>
<th>Remarks and examples</th>
<th>Also see</th>
</tr>
</thead>
</table>

**Description**

*paths* and the options above describe the model to be fit by *gsesm*.

**Syntax**

```
gsem paths ..., ... model_description_options
```

```
model_description_options          Description

family(), link(), ...              see [SEM] *gsesm family-and-link options*
* covariance()                     path notation for treatment of covariances;
                                          see [SEM] *sem and gsem path notation*
* variance()                       path notation for treatment of variances;
                                          see [SEM] *sem and gsem path notation*
* means()                          path notation for treatment of means;
                                          see [SEM] *sem and gsem path notation*
* covstructure()                   alternative method to place restrictions on covariances;
                                          see [SEM] *sem and gsem option covstructure()

collinear                          keep collinear variables
noconstant                         do not fit intercepts
naasis                             omit perfect predictors from Bernoulli models
noanchor                           do not apply default anchoring
forceranoanchor                    programmer’s option
* reliability()                   reliability of measurement variables;
                                          see [SEM] *sem and gsem option reliability()

consraints()                       specify constraints; see [SEM] *sem and gsem option constraints()
from()                             specify starting values; see [SEM] *sem and gsem option from()```

* Option may be specified more than once.

**Options**

*family()* and *link()* specify the distribution and link function, such as *family(poisson)* and *link(log)*, for generalized linear responses. There are lots of synonyms, so you can specify, for example, just *poisson*. In addition, there are *exposure()* and *offset()* options. See [SEM] *gsesm family-and-link options*.

*covariance()*., *variance()*., and *means()* fully describe the model to be fit. See [SEM] *sem and gsem path notation*. 

531
covstructure() provides a convenient way to constrain covariances in your model. Alternatively or in combination, you can place constraints by using the standard path notation. See [SEM] sem and gsem option covstructure().

collinear; see [R] Estimation options.

noconstant specifies that all intercepts be constrained to 0. See [SEM] sem and gsem path notation.

This option is seldom specified.

noasis specifies that perfect-predictor variables be omitted from all family Bernoulli models. By default, gsem does not omit the variable, so one can specify tricky models where an equation contains perfect predictors that are still identified through other portions of the model.

noanchor specifies that gsem not check for lack of identification or fill in anchors where needed. gsem is instead to issue an error message if anchors would be needed. Specify this option when you believe you have specified the necessary normalization constraints and want to hear about it if you are wrong. See Identification 2: Normalization constraints (anchoring) in [SEM] Intro 4.

forcenoanchor is similar to noanchor except that rather than issue an error message, gsem proceeds to estimation. There is no reason you should specify this option. forcenoanchor is used in testing of gsem at StataCorp.

reliability() specifies the fraction of variance not due to measurement error for a variable. See [SEM] sem and gsem option reliability().

constraints() specifies parameter constraints you wish to impose on your model; see [SEM] sem and gsem option constraints(). Constraints can also be specified as described in [SEM] sem and gsem path notation, and they are usually more conveniently specified using the path notation.

from() specifies the starting values to be used in the optimization process; see [SEM] sem and gsem option from(). Starting values can also be specified using the init() suboption as described in [SEM] sem and gsem path notation.

Remarks and examples

To use gsem successfully, you need to understand paths, covariance(), variance(), and means(); see Using path diagrams to specify standard linear SEMs in [SEM] Intro 2 and see [SEM] sem and gsem path notation.

covstructure() is often convenient; see [SEM] sem and gsem option covstructure().

Also see

[SEM] gsem — Generalized structural equation model estimation command
[SEM] Intro 2 — Learning the language: Path diagrams and command language
[SEM] sem and gsem option constraints() — Specifying constraints
[SEM] sem and gsem option covstructure() — Specifying covariance restrictions
[SEM] sem and gsem option from() — Specifying starting values
[SEM] sem and gsem option reliability() — Fraction of variance not due to measurement error
[SEM] sem and gsem path notation — Command syntax for path diagrams
Title

gsem path notation extensions — Command syntax for path diagrams

Description

This entry concerns gsem only.

The command syntax for describing generalized SEMs is fully specified by paths, covariance(), variance(), covstructure(), and means(); see [SEM] sem and gsem path notation and [SEM] sem and gsem option covstructure().

With gsem, the notation is extended to allow for generalized linear response variables, multilevel latent variables, categorical latent variables, and comparisons of groups. That is the subject of this entry.

Syntax

\[
\text{gsem paths} \ldots [, \text{covariance()} \text{variance()} \text{means()} \text{group()} \text{lclass()}] \\
gsem paths \ldots [, \text{covstructure()} \text{means()} \text{group()} \text{lclass()}]
\]

paths specifies the direct paths between the variables of your model.

The model to be fit is fully described by paths, covariance(), variance(), covstructure(), and means().

The syntax of these elements is modified when the group() or lclass() option is specified.

Options

covariance(), variance(), and means() are described in [SEM] sem and gsem path notation.

covstructure() is described in [SEM] sem and gsem option covstructure().

group(varname) allows models specified with paths, covariance(), variance(), covstructure(), and means() to be automatically interacted with the groups defined by varname; see [SEM] Intro 6. The syntax of paths and the arguments of covariance(), variance(), covstructure(), and means() gain an extra syntactical piece when group() is specified.

lclass() allows models specified with paths, covariance(), variance(), and covstructure() to be automatically interacted with categorical latent variables; see [SEM] Intro 2. The syntax of paths and the arguments of covariance(), variance(), and covstructure() gain an extra syntactical piece when lclass() is specified.

Remarks and examples

Remarks are presented under the following headings:

- Specifying family and link
- Specifying multilevel nested latent variables
- Specifying multilevel crossed latent variables
- Specifying paths for a specific group
- Specifying paths for a specific latent class
- Specifying paths for a specific group and latent class

533
Specifying family and link

\texttt{gsem} fits not only linear models but also generalized linear models. There is a set of options for specifying the specific model to be fit. These options are known as family-and-link options, which include \textit{family()} and \textit{link()}, but those options are seldom used in favor of other family-and-link shorthand options such as \textit{logit}, which means \textit{family(bernoulli)} and \textit{link(logit)}. These options are explained in \cite{SEM} \texttt{gsem family-and-link options}.

In the command language, you can specify these options among the shared options at the end of a \texttt{gsem} command:

\begin{verbatim}
  . gsem ..., ... logit ...
\end{verbatim}

That is convenient but only if all the equations in the model are using the same specific response function. Many models include multiple equations with each using a different response function.

You can specify any of the family-and-link options within paths. For instance, typing

\begin{verbatim}
  . gsem (y <- x1 x2), logit
\end{verbatim}

has the same effect as typing

\begin{verbatim}
  . gsem (y <- x1 x2, logit)
\end{verbatim}

Thus you can type

\begin{verbatim}
  . gsem (y1 <- x1 L, logit) (y2 <- x2 L, poisson) ..., ...
\end{verbatim}

The y1 equation would be logit, and the y2 equation would be Poisson. If you wanted y2 to be linear regression, you could type

\begin{verbatim}
  . gsem (y1 <- x1 L, logit) (y2 <- x2 L, regress) ..., ...
\end{verbatim}

or you could be silent and let y2 default to linear regression,

\begin{verbatim}
  . gsem (y1 <- x1 L, logit) (y2 <- x2 L) ..., ...
\end{verbatim}

Specifying multilevel nested latent variables

Latent variables are indicated by a name in which at least the first letter is capitalized. This generic form of the name is often written \textit{Lname}.

In regular latent variables, which we will call level-1 latent variables, the unobserved values vary observation by observation. Level-1 latent variables are the more common kinds of latent variables.

\texttt{gsem} allows higher-level latent variables as well as the level-1 variables. Let’s consider three-level data: students at the observational level, teachers at the second level, and schools at the third. In these data, each observation is a student. We have data on students nested within teachers nested within schools.

Let’s assume that we correspondingly have three identification (ID) variables. We number the following list with the nesting level of the data:

1. Variable \texttt{school} contains a school ID number. If two observations have the same value of \texttt{school}, then both of those students attended the same school.
2. Variable \texttt{teacher} contains a teacher ID number, or it contains a teacher-within-school ID number. That is, we do not care whether different schools assigned teachers the same ID number. It will be sufficient for us that the ID number is unique within school.
1. Variable student contains a student ID number, or it contains a student-within-school ID number, or even a student-within-teacher-within-school ID number. That is, we do not care whether different observations have the same student ID as long as they have different teacher IDs or different school IDs.

Here is how you write latent variable names at each level of the model:

3. Level 3 is the school level. Latent variables are written as 
   \[ Lname[school] \]
   An example would be SchQuality[school].
   The unobserved values of \[ Lname[school] \] vary across schools and are constant within school.
   If \[ Lname[school] \] is endogenous, its error variable is \[ e.Lname[school] \].
   You must refer to \[ Lname[school] \] without omitting the [school] part. \[ Lname \] by itself looks like another latent variable to gsem.

2. Level 2 is the teacher-within-school level. Latent variables are written as
   \[ Lname[school>teacher] \] or \[ Lname[teacher<school] \]
   An example would be TeachQuality[school>teacher] or TeachQuality[teacher<school].
   To gsem, \[ Lname[school>teacher] \] and \[ Lname[teacher<school] \] mean the same thing. You can even refer to TeachQuality[school>teacher] in one place and refer to TeachQuality[teacher<school] in another, and there will be no confusion.
   The unobserved values of \[ Lname[school>teacher] \] vary across schools and teachers, and they are constant within teacher.
   If \[ Lname[school>teacher] \] is endogenous, its error variable is \[ e.Lname[school>teacher] \] or, equivalently, \[ e.Lname[teacher<school] \].

1. Level 1 is the student or observational level. Latent variables are written as
   \[ Lname[school>teacher>student] \] or \[ Lname[student<teacher<school] \] or \[ Lname \]
   Everybody just writes \[ Lname \]. These are the latent variables that correspond to the latent variables that sem provides. Unobserved values within the latent variable vary observation by observation.
   If \[ Lname \] is endogenous, its error variable is \[ e.Lname \].

You can use multilevel latent variables in paths and options just as you would use any other latent variable; see [SEM] sem and gsem path notation. Remember, however, that you must type out the full name of all but the first-level latent variables. You type, for instance, SchQual[school>teacher]. There is a real tendency to type just SchQual when the name is unique.

Changing the subject, we see that the names by which effects are referred to are a function of the top level. We just discussed a three-level model. The three levels of the model were

(3) school
(2) school>teacher
(1) school>teacher>student
If we had a two-level model, the levels would be

(2) teacher
(1) teacher>student

Thus, if we had started with a two-level model and then wanted to add a third, higher level onto it, latent variables that were previously referred to as, say, TeachQual[teacher] would now be referred to as TeachQual[school>teacher].

**Specifying multilevel crossed latent variables**

In our previous example, we had a three-level nested model in which student was nested within teacher, which was nested within school.

Let’s consider data on employees that also have the characteristics of working in an occupation and working in an industry. These variables are not nested. Just as before, we will assume we have variable employee containing an employee ID and variables industry and occupation. The latent variables associated with this model could be the following:

<table>
<thead>
<tr>
<th>Level</th>
<th>Latent-variable name</th>
</tr>
</thead>
<tbody>
<tr>
<td>occupation</td>
<td>Lname[occupation]</td>
</tr>
<tr>
<td>industry</td>
<td>Lname[industry]</td>
</tr>
<tr>
<td>employee (observational)</td>
<td>Lname</td>
</tr>
</tbody>
</table>

**Specifying paths for a specific group**

The group(varname) option,

```
gsem ..., ... group(varname)
```

specifies that the model be fit separately for the different values of varname. varname might be sex, and then the model would be fit separately for males and females, or varname might be something else and perhaps take on more than two values.

Whatever varname is, group(varname) defaults to letting some of the path coefficients, covariances, variances, and means of your model vary across the groups and constraining others to be equal. Which parameters vary and which are constrained is described in [SEM] gsem group options, but that is a minor detail right now.

In what follows, we will assume that varname is mygrp and takes on three values. Those values are 1, 2, and 3, but they could just as well be 2, 9, and 12.

Consider typing

```
gsem ..., ...
```

and typing

```
gsem ..., ... group(mygrp)
```

Whatever paths, covariance(), variance(), covstructure(), and means() are that describe the model, there are now three times as many parameters because each group has its own unique set. In fact, when you give the second command, you are not merely asking for three times the parameters, you are specifying three models, one for each group! In this case, you specified the same model three times without knowing it.
You can vary the model specified across groups.

1. Let’s write the model you wish to fit as

   \[ \texttt{. gsem (a) (b) (c), cov(d) cov(e) var(f)} \]

   where \( a, b, \ldots, f \) stand for what you type. In this generic example, we have two \texttt{cov()}\ options just because multiple \texttt{cov()}\ options often occur in real models. When you type

   \[ \texttt{. gsem (a) (b) (c), cov(d) cov(e) var(f) group(mygrp)} \]

   results are as if you typed

   \[ \texttt{. gsem (1: a) (2: a) (3: a)} \\
   \texttt{(1: b) (2: b) (3: b)} \\
   \texttt{(1: c) (2: c) (3: c),} \\
   \texttt{cov(1: d) cov(2: d) cov(3: d)} \\
   \texttt{cov(1: e) cov(2: e) cov(3: e)} \\
   \texttt{var(1: f) cov(2: f) cov(3: f) group(mygrp)} \]

   The 1:, 2:, and 3: identify the groups for which paths, covariances, or variances are being added, modified, or constrained.

   If \texttt{mygrp} contained the unique values 5, 8, and 10 instead of 1, 2, and 3, then 5: would appear in place of 1:, 8: would appear in place of 2:, and 10: would appear in place of 3::.

2. Consider the model

   \[ \texttt{. gsem (y <- x) (b) (c), cov(d) cov(e) var(f) group(mygrp)} \]

   The default \texttt{ginvariant()}\ option constrains all intercepts, coefficients, and loadings to be the same across all groups. If you wanted to constrain the path coefficient \((y <- x)\) to be the same across all three groups, but let all other parameters be group specific, you could type

   \[ \texttt{. gsem (y <- x@c1) (b) (c), cov(d) cov(e) var(f) group(mygrp) ginvariant(none)} \]

   This works because the expansion of \((y <- x@c1)\) is

   \[ (1: y <- x@c1) (2: y <- x@c1) (3: y <- x@c1) \]

   See item 12 in \texttt{SEM} for more examples of specifying constraints.

3. Consider the model

   \[ \texttt{. gsem (y <- x) (b) (c), cov(d) cov(e) var(f) group(mygrp)} \]

   If you wanted to constrain the path coefficient \((y <- x)\) to be the same in groups 2 and 3, you could type

   \[ \texttt{. gsem (1: y <- x) (2: y <- x@c1) (3: y <- x@c1) (b) (c),} \\
   \texttt{cov(d) cov(e) var(f) group(mygrp)} \]

   The default \texttt{ginvariant()}\ option still constrains the other coefficients to be equal to each other, but not necessarily equal to the coefficient in groups 2 and 3. In our example, \texttt{mygrp} has 3 levels, so the above effectively removed the default constraint on the group 1 coefficient.

4. Instead of following item 3, you could type

   \[ \texttt{. gsem (y <- x) (2: y <- x@c1) (3: y <- x@c1) (b) (c),} \\
   \texttt{cov(d) cov(e) var(f) group(mygrp)} \]

   The part \((y <- x) (2: y <- x@c1) (3: y <- x@c1)\) expands to

   \[ (1: y <- x) (2: y <- x) (3: y <- x) (2: y <- x@c1) (3: y <- x@c1) \]
and thus the path is defined twice for group 2 and twice for group 3. When a path is defined more than once, the definitions are combined. In this case, the second definition adds more information, so the result is as if you typed

(1: y <- x) (2: y <- x@c1) (3: y <- x@c1)

5. Instead of following item 3 or item 4, you could type

```bash
.gsem (y <- x@c1) (1: y <- x@c2) (b) (c), ///
    cov(d) cov(e) var(f) group(mygrp)
```

The part (y <- x@c1) (1: y <- x@c2) expands to

(1: y <- x@c1) (2: y <- x@c1) (3: y <- x@c1) (1: y <- x@c2)

When results are combined from repeated definitions, then definitions that appear later take precedence. In this case, results are as if the expansion read

(1: y <- x@c2) (2: y <- x@c1) (3: y <- x@c1)

Thus coefficients for groups 2 and 3 are constrained. The group-1 coefficient is constrained to c2. If c2 appears nowhere else in the model specification, then results are as if the path for group 1 were unconstrained.

6. Instead of following item 3, item 4, or item 5, you could not type

```bash
.gsem (y <- x@c1) (1: y <- x) (b) (c), ///
    cov(d) cov(e) var(f) group(mygrp)
```

The expansion of (y <- x@c1) (1: y <- x) reads

(1: y <- x@c1) (2: y <- x@c1) (3: y <- x@c1) (1: y <- x)

and you might think that 1: y <- x would replace 1: y <- x@c1. Information, however, is combined, and even though precedence is given to information appearing later, silence does not count as information. Thus the expanded and reduced specification reads the same as if 1: y <- x was never specified:

(1: y <- x@c1) (2: y <- x@c1) (3: y <- x@c1)

7. Items 1–6, stated in terms of paths, apply equally to what is typed inside the `means()`, `variance()`, `covariance()`, and `covstructure()` options. For instance, if you typed

```bash
.gsem (a) (b) (c), var(e.y@c1) group(mygrp)
```

then you are constraining the variance to be equal across all three groups.

If you wanted to constrain the variance to be equal in groups 2 and 3, you could type

```bash
.gsem (a) (b) (c), var(e.y) var(2: e.y@c1) var(3: e.y@c1), group(mygrp)
```

You could omit typing `var(e.y)` because it is implied. Alternatively, you could type

```bash
.gsem (a) (b) (c), var(e.y@c1) var(1: e.y@c2) group(mygrp)
```
You could not type
\[
gsem (a) (b) (c), \var(e.y@c1) \var(1: e.y) \text{ group(mygrp)}
\]
because silence does not count as information when specifications are combined.

Similarly, if you typed
\[
gsem (a) (b) (c), \cov(e.y1*e.y2@c1) \text{ group(mygrp)}
\]
than you are constraining the covariance to be equal across all groups. If you wanted to constrain
the covariance to be equal in groups 2 and 3, you could type
\[
gsem (a) (b) (c), \cov(e.y1*e.y2) ///
\begin{align*}
&\cov(2: e.y1*e.y2@c2) \\
&\cov(3: e.y1*e.y2@c3) ///
\end{align*}
\text{ group(mygrp)}
\]
You could not omit \(\cov(e.y1*e.y2)\) because it is not assumed. By default, error variables
are assumed to be uncorrelated. Omitting the option would constrain the covariance to be 0 in
group 1 and to be equal in groups 2 and 3.

Alternatively, you could type
\[
gsem (a) (b) (c), \cov(e.y1*e.y2@c1) ///
\begin{align*}
&\cov(1: e.y1*e.y2@c2) ///
&\text{ group(mygrp)}
\end{align*}
\]

8. In the examples above, we have referred to the groups with their numeric values, 1, 2, and 3.
Had the values been 5, 8, and 10, then we would have used those values.

If the group variable \text{mygrp} has a value label, you can use the label to refer to the group. For
instance, imagine \text{mygrp} is labeled as follows:
\[
\begin{align*}
\text{label define grpvals 1 Male 2 Female 3 "Unknown sex"} \\
\text{label values mygrp grpvals}
\end{align*}
\]

We could type
\[
\begin{align*}
\text{gsem (y <- x) (Female: y <- x@c1) (Unknown sex: y <- x@c1) \ldots, \ldots}
\end{align*}
\]
or we could type
\[
\begin{align*}
\text{gsem (y <- x) (2: y <- x@c1) (3: y <- x@c1) \ldots, \ldots}
\end{align*}
\]

**Specifying paths for a specific latent class**

The \text{lclass()} option in
\[
\text{gsem \ldots, \ldots lclass(C 3)}
\]
specifies that the model be fit separately for each of 3 classes of the categorical latent variable \text{C}. The
classes of \text{C} are 1, 2, and 3. \text{gsem} allows for more than one categorical latent variable to be specified
by allowing more than one \text{lclass()} option. The two \text{lclass()} options in
\[
\text{gsem \ldots, \ldots lclass(C 3) lclass(D 2)}
\]
specify that the model be fit separately for each of the 6 latent classes defined by the interaction
between the categorical latent variables \text{C} and \text{D}.

Whatever the number of classes and latent variables, \text{gsem} defaults to letting some of the path
coefficients, covariances, and variances of your model to vary across the latent classes and constraining
others to be equal. Which parameters vary and which are constrained is described in [SEM] \text{gsem lclass options},
but that is a minor detail right now.
Consider typing

```
    . gsem ..., ...
```

then typing

```
    . gsem ..., ... lclass(C 3)
```

Whatever paths, covariance(), variance(), and covstructure() are that describe the model, there are now three times as many parameters because each latent class has its own unique set. In fact, when you give the second command, you are not merely asking for three times the parameters, you are specifying three models, one for each latent class, and a multinomial model for latent class probabilities.

You can vary the model specified across latent classes.

1. Let’s write the model you wish to fit as

```
  . gsem (a) (b) (c), cov(d) cov(e) var(f)
```

where \(a, b, \ldots, f\) stand for what you type. In this generic example, we have two \(\text{cov()}\) options just because multiple \(\text{cov()}\) options can occur in real models. When you type

```
  . gsem (a) (b) (c), cov(d) cov(e) var(f) lclass(C 3)
```

results are as if you typed

```
  . gsem (1: a) (2: a) (3: a) ///
  (1: b) (2: b) (3: b) ///
  (1: c) (2: c) (3: c) ///
  (1.C <- _cons@0) ///
  (2.C <- _cons) ///
  (3.C <- _cons), ///
  cov(1: d) cov(2: d) cov(3: d) ///
  cov(1: e) cov(2: e) cov(3: e) ///
  var(1: f) cov(2: f) cov(3: f) ///
  lclass(C 3)
```

The 1:, 2:, and 3: identify the latent classes for which paths, covariances, or variances are being added, modified, or constrained.

The paths to 1.C, 2.C, and 3.C identify linear predictions for the multinomial probabilities for the categorical latent variable \(C\).

2. Consider the model

```
  . gsem (y <- x) (b) (c), cov(d) cov(e) var(f) lclass(C 3)
```

If you wanted to constrain the path coefficient (\(y \leftarrow x\)) to be the same across all three latent classes, you could type

```
  . gsem (y <- x@c1) (b) (c), cov(d) cov(e) var(f) lclass(C 3)
```

This works because the expansion of \((y \leftarrow x@c1)\) is

\[(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\]

See item 12 in [SEM] sem and gsem path notation for more examples of specifying constraints.

3. Consider the model

```
  . gsem (y <- x) (b) (c), cov(d) cov(e) var(f) lclass(C 3)
```
If you wanted to constrain the path coefficient \((y \leftarrow x)\) to be the same in latent classes 2 and 3, you could type

```
.gsem (1: y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (b) (c), ///
       cov(d) cov(e) var(f) lclass(C 3)
```

4. Instead of following item 3, you could type

```
.gsem (y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (b) (c), ///
       cov(d) cov(e) var(f) lclass(C 3)
```

The part \((y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\) expands to

\[(1: y \leftarrow x) (2: y \leftarrow x) (3: y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\]

and thus the path is defined twice for latent class 2 and twice for latent class 3. When a path is defined more than once, the definitions are combined. In this case, the second definition adds more information, so the result is as if you typed

\[(1: y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\]

5. Instead of following item 3 or item 4, you could type

```
.gsem (y \leftarrow x@c1) (1: y \leftarrow x@c2) (b) (c), ///
       cov(d) cov(e) var(f) lclass(C 3)
```

The part \((y \leftarrow x@c1) (1: y \leftarrow x@c2)\) expands to

\[(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (1: y \leftarrow x@c2)\]

When results are combined from repeated definitions, then definitions that appear later take precedence. In this case, results are as if the expansion read

\[(1: y \leftarrow x@c2) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\]

Thus coefficients for latent classes 2 and 3 are constrained. The first latent class coefficient is constrained to \(c2\). If \(c2\) appears nowhere else in the model specification, then results are as if the path for the first latent class were unconstrained.

6. Instead of following item 3, item 4, or item 5, you could not type

```
.gsem (y \leftarrow x@c1) (1: y \leftarrow x) (b) (c), ///
       cov(d) cov(e) var(f) lclass(C 3)
```

The expansion of \((y \leftarrow x@c1) (1: y \leftarrow x)\) reads

\[(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (1: y \leftarrow x)\]

and you might think that \(1: y \leftarrow x\) would replace \(1: y \leftarrow x@c1\). Information, however, is combined, and even though precedence is given to information appearing later, silence does not count as information. Thus the expanded and reduced specification reads the same as if \(1: y \leftarrow x\) was never specified:

\[(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\]

7. Items 1–6, stated in terms of paths, apply equally to what is typed inside the variance(), covariance(), and covstructure() options. For instance, if you typed

```
.gsem (a) (b) (c), var(e.y@c1) lclass(C 3) lcinvariant(none)
```

then you are constraining the error variance of \(y\) to be equal across all three latent classes. Without the \(lcinvariant(none)\) option, `gsem` would constrain all error variances to be equal across all groups.

If you wanted to constrain the variance to be equal in latent classes 2 and 3, you could type

```
.gsem (a) (b) (c), var(e.y) var(2: e.y@c1) var(3: e.y@c1), lclass(C 3)
```
You could omit typing \texttt{var(e.y)} because it is implied. Alternatively, you could type
\begin{verbatim}
. gsem (a) (b) (c), var(e.y@c1) var(1: e.y@c2) lclass(C 3)
\end{verbatim}
You could not type
\begin{verbatim}
. gsem (a) (b) (c), var(e.y@c1) var(1: e.y) lclass(C 3)
\end{verbatim}
because silence does not count as information when specifications are combined.
Similarly, if you typed
\begin{verbatim}
. gsem (a) (b) (c), cov(e.y1*e.y2@c1) lclass(C 3)
\end{verbatim}
then you are constraining the covariance to be equal across all latent classes. If you wanted to constrain the covariance to be equal in latent classes 2 and 3, you could type
\begin{verbatim}
. gsem (a) (b) (c), cov(e.y1*e.y2) ///
   cov(2: e.y1*e.y2@c1) cov(3: e.y1*e.y2@c1) ///
   lclass(C 3)
\end{verbatim}
You could not omit \texttt{cov(e.y1*e.y2)} because it is not assumed. By default, error variables are assumed to be uncorrelated. Omitting the option would constrain the covariance to be 0 in latent class 1 and to be equal in latent classes 2 and 3.
Alternatively, you could type
\begin{verbatim}
. gsem (a) (b) (c), cov(e.y1*e.y2@c1) cov(1: e.y1*e.y2@c2) lclass(C 3)
\end{verbatim}
8. The above examples focused on models with a single categorical latent variable. The same rules apply for models with two or more categorical latent variables, with the added requirement that the latent class prefix is specified using factor-variable notation.

For example, suppose your model is
\begin{verbatim}
. gsem (a), cov(b) var(c) lclass(C 3) lclass(D 2)
\end{verbatim}
The result is as if you typed
\begin{verbatim}
   (1.C <- _cons@0) (2.C <- _cons) (3.C <- _cons) ///
   (1.D <- _cons@0) (2.D <- _cons), ///
     cov(1.C#1.D: b) cov(1.C#2.D: b) ///
     cov(2.C#1.D: b) cov(2.C#2.D: b) ///
     cov(3.C#1.D: b) cov(3.C#2.D: b) ///
     var(1.C#1.D: c) var(1.C#2.D: c) ///
     var(2.C#1.D: c) var(2.C#2.D: c) ///
     var(3.C#1.D: c) var(3.C#2.D: c) ///
   lclass(C 3) lclass(D 2)
\end{verbatim}
Each of the \texttt{paths}, \texttt{variance()}, \texttt{covariance()}, and \texttt{covstructure()} options is now targeted to a specific latent class.

9. From the above example, we see that the multinomial probabilities for the latent classes are modeled as if \texttt{C} and \texttt{D} were independent. For example, the linear prediction for the latent class \texttt{3.C#2.D} is composed of the intercept for \texttt{3.C} plus the intercept for \texttt{2.D}.

Add the path \texttt{3.C#2.D <- _cons} to the above model specification, and the linear prediction for the latent class \texttt{3.C#2.D} will be composed of the intercept for \texttt{3.C}, plus the intercept for \texttt{2.D}, plus the new specified intercept for \texttt{3.C#2.D}. With this new intercept, \texttt{C} and \texttt{D} are no longer modeled independently.
10. Consider the model
   . gsem (y <- x), lclass(C 3) lclass(D 2)

   If you wanted to constrain the path coefficient \((y <- x)\) to be the same in all latent classes with
1.C, then add the path \((1.C: y <- x@c1)\) so that the model is
   . gsem (y <- x) (1.C: y <- x@c1), lclass(C 3) lclass(D 2)

   This essentially expands to
   . gsem (1.C#1.D: y <- x@c1) (1.C#2.D: y <- x@c1) ///
     (2.C#1.D: y <- x) (2.C#2.D: y <- x) ///
     (3.C#1.D: y <- x) (3.C#2.D: y <- x) ///
     (1.C <- _cons@0) (2.C <- _cons) (3.C <- _cons) ///
     (1.D <- _cons@0) (2.D <- _cons), ///
     lclass(C 3) lclass(D 2)

   This shortcut syntax works similarly for options covariance(), variance(), and covstructure().

Specifying paths for a specific group and latent class

gsem allows models where both option group() and option lclass() are specified. Building on
the examples from the above sections, the model
   . gsem ..., ... group(mygrp) lclass(D 2)
specifies that the model with categorical latent variable D be fit separately for each level of mygrp. The number of model parameters multiplies when options group() and lclass() are specified.

   For example, the model
   . gsem (y <- x), group(mygrp) lclass(D 2)

   with mygrp having levels 1, 2, and 3, is equivalent to
   . gsem (1.mygrp#1.D: y <- x) (1.mygrp#2.D: y <- x) ///
     (2.mygrp#1.D: y <- x) (2.mygrp#2.D: y <- x) ///
     (3.mygrp#1.D: y <- x) (3.mygrp#2.D: y <- x) ///
     (1.mygrp: 1.D <- _cons@0) (1.mygrp: 2.D <- _cons) ///
     (2.mygrp: 1.D <- _cons@0) (2.mygrp: 2.D <- _cons) ///
     (3.mygrp: 1.D <- _cons@0) (3.mygrp: 2.D <- _cons), ///
     group(mygrp) lclass(D 2)

   The path \((y <- x)\) expands to 6 paths, one for each group and latent class combination. The number of covariances and variances is similarly multiplied by 6. The linear predictions for D expand by the number of group levels.

Also see

[SEM] gsem — Generalized structural equation model estimation command

[SEM] gsem group options — Fitting models on different groups

[SEM] gsem lclass options — Fitting models with latent classes

[SEM] sem and gsem path notation — Command syntax for path diagrams

[SEM] Intro 2 — Learning the language: Path diagrams and command language
Postestimation commands

The following are the postestimation commands that you can use after estimation by \texttt{gsem}:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{estat eform}</td>
<td>display exponentiated coefficients</td>
</tr>
<tr>
<td>\texttt{estat ic}</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>\texttt{estat lcgof}</td>
<td>latent class goodness-of-fit statistic</td>
</tr>
<tr>
<td>\texttt{estat lcmean}</td>
<td>latent class marginal means</td>
</tr>
<tr>
<td>\texttt{estat lcprob}</td>
<td>latent class marginal probabilities</td>
</tr>
<tr>
<td>\texttt{estat sd}</td>
<td>display variance components as standard deviations and correlations</td>
</tr>
<tr>
<td>\texttt{hausman}</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>\texttt{lrtest}</td>
<td>likelihood-ratio tests</td>
</tr>
<tr>
<td>\texttt{test}</td>
<td>Wald tests</td>
</tr>
<tr>
<td>\texttt{lincom}</td>
<td>linear combination of parameters</td>
</tr>
<tr>
<td>\texttt{nlcom}</td>
<td>nonlinear combination of parameters</td>
</tr>
<tr>
<td>\texttt{testnl}</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
<tr>
<td>\texttt{estat summarize}</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>\texttt{estat vce}</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>\texttt{predict}</td>
<td>generalized linear predictions, etc.</td>
</tr>
<tr>
<td>\texttt{predictnl}</td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td>\texttt{margins}</td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td>\texttt{marginsplot}</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>\texttt{contrast}</td>
<td>contrasts and linear hypothesis tests</td>
</tr>
<tr>
<td>\texttt{pwcompare}</td>
<td>pairwise comparisons</td>
</tr>
<tr>
<td>\texttt{estimates}</td>
<td>cataloging estimation results</td>
</tr>
</tbody>
</table>

\texttt{hausman} and \texttt{lrtest} are not appropriate with \texttt{svy} estimation results.

For a summary of postestimation features, see \texttt{[SEM] Intro 7}.

Postestimation commands such \texttt{lincom} and \texttt{nlcom} require referencing estimated parameter values, which are accessible via \texttt{_b[name]}. To find out what the names are, type \texttt{sem, coeflegend}.

544
margins

Description for margins

margins estimates margins of response for expected values, probabilities, and predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options ]
margins [marginlist], predict(statistic ...)[predict(statistic ...)] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>calculate expected values for each depvar</td>
</tr>
<tr>
<td>mu</td>
<td>calculate expected value of depvar</td>
</tr>
<tr>
<td>pr</td>
<td>calculate probability (synonym for mu when ( \mu ) is a probability)</td>
</tr>
<tr>
<td>eta</td>
<td>calculate expected value of linear prediction of depvar</td>
</tr>
<tr>
<td>expression(exp)</td>
<td>calculate expected prediction using exp</td>
</tr>
<tr>
<td>classpr</td>
<td>calculate latent class probabilities</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>survival</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>latent</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>latent(varlist)</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>classposteriorpr</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

mu defaults to the first depvar if option outcome() is not specified. If depvar is family(multinomial) or family(ordinal) the default is the first level of the outcome.

pr defaults to the first depvar that allows predicted probabilities if option outcome() is not specified. If depvar is family(multinomial) or family(ordinal) the default is the first level of the outcome.

eta defaults to the first depvar if option outcome() is not specified. If depvar is family(multinomial) the default is the first level of the outcome.

classpr defaults to the first latent class if option class() is not specified.

predict’s option marginal is assumed if predict’s options conditional(fixedonly) and class() are not specified; see [SEM] predict after gsem.

Statistics not allowed with margins are functions of stochastic quantities other than e(b). For the full syntax, see [R] margins.

Remarks and examples

This manual entry concerns gsem. For information on postestimation features available after sem, see [SEM] sem postestimation.
Also see

[SEM] gsem reporting options — Options affecting reporting of results
[U] 20 Estimation and postestimation commands
**gsem reporting options — Options affecting reporting of results**

### Description

These options control how `gsem` displays estimation results.

### Syntax

```plaintext
gsem paths ..., ... reporting_options
```

```plaintext
gsem, reporting_options
```

<table>
<thead>
<tr>
<th><code>reporting_options</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>coeflegend</code></td>
<td>display coefficient legend</td>
</tr>
<tr>
<td><code>nocnsreport</code></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><code>noheader</code></td>
<td>do not display header above parameter table</td>
</tr>
<tr>
<td><code>nodvheader</code></td>
<td>do not display dependent variables information in the header</td>
</tr>
<tr>
<td><code>notable</code></td>
<td>do not display parameter table</td>
</tr>
<tr>
<td><code>byparm</code></td>
<td>display results in a single table with rows arranged by parameter</td>
</tr>
</tbody>
</table>

| `display_options` | |
|-------------------| control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling |

### Options

`level(#); see [R] Estimation options.`

- `coeflegend` displays the legend that reveals how to specify estimated coefficients in `_b[]` notation, which you are sometimes required to type in specifying postestimation commands.

- `nocnsreport` suppresses the display of the constraints. Fixed-to-zero constraints that are automatically set by `gsem` are not shown in the report to keep the output manageable.

- `noheader` suppresses the header above the parameter table, the display that reports the final log-likelihood value, number of observations, etc.

- `nodvheader` suppresses the dependent variables information from the header above the parameter table.

- `notable` suppresses the parameter table.

- `byparm` specifies that estimation results with multiple groups or latent classes be reported in a single table with rows arranged by parameter. The default is to report results in separate tables for each group and latent class combination.
display options: noic, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] Estimation options.

Remarks and examples

Any of the above options may be specified when you fit the model or when you redisplay results, which you do by specifying nothing but options after the gsem command:

```
. gsem (...) (...), ...
    (original output displayed)
. gsem
    (output redisplayed)
. gsem, coeflegend
    (coefficient-name table displayed)
. gsem
    (output redisplayed)
```

Also see

[SEM] gsem — Generalized structural equation model estimation command

[SEM] Example 29g — Two-parameter logistic IRT model
**Description**

`lincom` is a postestimation command for use after `sem`, `gsem`, and nearly all Stata estimation commands.

`lincom` computes point estimates, standard errors, $z$ statistics, $p$-values, and confidence intervals for linear combinations of the estimated parameters.

After `sem` and `gsem`, you must use the `_b[ ]` coefficient notation; you cannot refer to variables by using shortcuts to obtain coefficients on variables.

See [R] `lincom`.

**Menu**

Statistics $>$ SEM (structural equation modeling) $>$ Testing and CIs $>$ Linear combinations of parameters

**Syntax**

```
lincom exp [ , options ]
```

**Options**

See *Options* in [R] `lincom`.

**Remarks and examples**

`lincom` works in the metric of SEM, which is to say path coefficients, variances, and covariances. If you want to frame your linear combinations in terms of standardized coefficients and correlations, and you fit the model with `sem`, not `gsem`, then prefix `lincom` with `estat stdize`; see [SEM] `estat stdize`.

**Stored results**

See *Stored results* in [R] `lincom`.

**Also see**

[R] `lincom` — Linear combinations of parameters

[SEM] `estat stdize` — Test standardized parameters

[SEM] `nlcom` — Nonlinear combinations of parameters

[SEM] `test` — Wald test of linear hypotheses
Description

`lrtest` is a postestimation command for use after `sem`, `gsem`, and other Stata estimation commands. `lrtest` performs a likelihood-ratio test comparing models. See [R] `lrtest`.

Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Likelihood-ratio test

Syntax

```
{sem|gsem} ... , ...                              (fit model 1)
estimates store `modelname1'

{sem|gsem} ... , ...                              (fit model 2)
estimates store `modelname2'

lrtest `modelname1' `modelname2'                     (compare models)
```

where one of the models is constrained and the other is unconstrained. `lrtest` counts parameters to determine which model is constrained and which is unconstrained, so it does not matter which model is which.

Warning concerning use with `sem`: Do not omit variables, observed or latent, from the model. Constrain their coefficients to be 0 instead. The models being compared must contain the same set of variables. This is because the standard SEM likelihood value is a function of the variables appearing in the model. Despite this fact, use of `lrtest` is appropriate under the relaxed conditional normality assumption.

Note concerning `gsem`: The above warning does not apply to `gsem` just as it does not apply to other Stata estimation commands. Whether you omit variables or constrain coefficients to 0, results will be the same. The generalized SEM likelihood is conditional on the exogenous variables.

Remarks and examples

See [SEM] Example 10 and [SEM] Example 39g.

When using `lrtest` after `sem`, you must be careful that the models being compared have the same observed and latent variables. For instance, the following is allowed:
lrtest — Likelihood-ratio test of linear hypothesis

. sem (L1 -> x1 x2 x3) (L1 <- x4 x5) (x1 <- x4) (x2 <- x5)
. estimates store m1
. sem (L1 -> x1 x2 x3) (L1 <- x4 x5)
. estimates store m2
. lrtest m1 m2

This is allowed because both models contain the same variables, namely, L1, x1, . . . , x5, even though the second model omitted some paths.

The following would produce invalid results:

. sem (L1 -> x1 x2 x3) (L1 <- x4 x5) (x1 <- x4) (x2 <- x5)
. estimates store m1
. sem (L1 -> x1 x2 x3) (L1 <- x4)
. estimates store m2
. lrtest m1 m2

It produces invalid results because the second model does not include variable x5 and the first model does. To run this test correctly, you type

. sem (L1 -> x1 x2 x3) (L1 <- x4 x5) (x1 <- x4) (x2 <- x5)
. estimates store m1
. sem (L1 -> x1 x2 x3) (L1 <- x4@0)
. estimates store m2
. lrtest m1 m2

If we were using gsem rather than sem, all the above would still be valid.

Stored results

See Stored results in [R] lrtest.

Also see

[SEM] Example 10 — MIMIC model
[SEM] Example 39g — Three-level model (multilevel, generalized response)
[R] lrtest — Likelihood-ratio test after estimation
[SEM] test — Wald test of linear hypotheses
[SEM] estat stdize — Test standardized parameters
[SEM] estat eqtest — Equation-level tests that all coefficients are zero
Methods and formulas for gsem — Methods and formulas for gsem

Description

The methods and formulas for the gsem command are presented below.

Remarks and examples

Remarks are presented under the following headings:

- Introduction
- Families of distributions
  - The Bernoulli family
  - The beta family
  - The binomial family
  - The ordinal family
  - The multinomial family
  - The Poisson family
  - The negative binomial family
  - The Gaussian family
  - Reliability
  - Point mass
- Link functions
  - The logit link
  - The probit link
  - The complementary log-log link
  - The log link
  - The identity link
- Survival distributions
  - The exponential distribution
  - The Weibull distribution
  - The gamma distribution
  - The loglogistic distribution
  - The lognormal distribution
- Models with continuous latent variables
  - The likelihood
  - Gauss–Hermite quadrature
  - Adaptive quadrature
  - Laplacian approximation
  - Survey data
  - Predictions
    - Empirical Bayes
    - Other predictions
- Models with categorical latent variables
  - The likelihood
  - The EM algorithm
  - Survey data
  - Predictions
Introduction

\texttt{gsem} fits generalized linear models with categorical or continuous latent variables via maximum likelihood. Here is a table identifying the family/link combinations that \texttt{gsem} allows.

<table>
<thead>
<tr>
<th>Family</th>
<th>logit</th>
<th>probit</th>
<th>cloglog</th>
<th>log</th>
<th>identity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bernoulli</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>binomial</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ordinal</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>multinomial</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>negative binomial</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Gaussian</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>exponential</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weibull</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gamma</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>loglogistic</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lognormal</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pointmass</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Log-likelihood calculations for fitting any model with continuous latent variables require integrating out the latent variables. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. \texttt{gsem} implements four different methods for numerically evaluating the integrals.

1. Gauss–Hermite quadrature (GHQ)
2. Mean-variance adaptive quadrature (MVAGH)
3. Mode-curvature adaptive quadrature (MCAGH)
4. Laplacian approximation

The default method is MVAGH. The numerical integration method for MVAGH is based on Rabe-Hesketh, Skrondal, and Pickles (2005), and the other numerical integration methods described in this manual entry are based on Skrondal and Rabe-Hesketh (2004, chap. 6.3).

Families of distributions

\texttt{gsem} implements the most commonly used distribution families associated with generalized linear models. \texttt{gsem} also implements distributions for ordinal and multinomial outcomes.

In this manual entry, observed endogenous variables are also known as generalized responses or generalized outcomes, but we will simply refer to them as responses or outcomes. The random variable corresponding to a given response will be denoted by $Y$. An observed value of $Y$ will be denoted by $y$, and the expected value of $Y$ by $\mu$. For the ordinal and multinomial families, we will refer to a linear prediction, denoted by $z$, instead of the expected value.
The Bernoulli family

The Bernoulli family is a binary response model. The response $Y$ is assumed to take on the values 0 or 1; however, \texttt{gsem} allows any nonzero and nonmissing value to mean 1.

The log of the conditional probability mass function is

$$
\log f(y|\mu) = y \log \mu + (1 - y) \log(1 - \mu)
$$

where $\mu$ is also known as the probability of a success. The default link for the Bernoulli family is the logit link.

The beta family

The beta family is a fractional response model. The response $Y$ is assumed to take on the real values between 0 and 1.

The log of the conditional probability density function is

$$
\log f(y|\mu, s) = \log \left\{ \Gamma(s) \right\} - \log \left\{ \Gamma(\mu s) \right\} - \log \left\{ \Gamma(s - \mu s) \right\}
+ (\mu - 1) \log y + (s - \mu s - 1) \log(1 - y)
$$

where $\mu$ is the mean of $Y$, and $s$ is the scale parameter. \texttt{gsem} fits $s$ in the log scale. The default link for the beta family is the logit link.

The binomial family

The binomial family is a count response model and generalizes the Bernoulli family by taking the sum of $k$ independent Bernoulli outcomes. The response $Y$ is assumed to take on the values 0, 1, \ldots, $k$.

The log of the conditional probability mass function is

$$
\log f(y|\mu) = \log \left\{ \Gamma(k + 1) \right\} - \log \left\{ \Gamma(y + 1) \right\} - \log \left\{ \Gamma(k - y + 1) \right\}
+ y \log \mu + (1 - y) \log(1 - \mu)
$$

where $\mu$ is the expected value for a single Bernoulli outcome. The default link for the binomial family is the logit link.

The ordinal family

The ordinal family is a discrete response model. The response $Y$ is assumed to take on one of $k$ unique values. The actual values are irrelevant except that higher values are assumed to correspond to “higher” outcomes. Without loss of generality, we will assume that $Y$ takes on the values 1, \ldots, $k$.

The ordinal family with $k$ outcomes has cutpoints $\kappa_0, \kappa_1, \ldots, \kappa_k$, where $\kappa_0 = -\infty$, $\kappa_y < \kappa_{y+1}$, and $\kappa_k = +\infty$.

Given a linear prediction $z$, the probability that a random response $Y$ takes the value $y$ is

$$
\Pr(Y = y|z) = \Pr(Y^* < \kappa_y - z) - \Pr(Y^* < \kappa_{y-1} - z)
$$
where \( Y^* \) is the underlying stochastic component for \( Y \). The distribution for \( Y^* \) is determined by the link function. \texttt{gsem} allows \texttt{logit}, \texttt{probit}, and \texttt{cloglog} for the ordinal family. The \texttt{logit} link assigns \( Y^* \) the extreme value distribution that is synonymous with the logit link for Bernoulli outcomes. The \texttt{probit} link assigns \( Y^* \) the standard normal distribution that is synonymous with the probit link for Bernoulli outcomes. The \texttt{cloglog} link assigns \( Y^* \) the distribution that is synonymous with the complementary log-log link for Bernoulli outcomes. The default link for the ordinal family is the \texttt{logit} link.

**The multinomial family**

The multinomial family is a discrete response model. The response \( Y \) is assumed to take on one of \( k \) unique values. The actual values are irrelevant and order does not matter; however, \texttt{gsem} requires that the values are nonnegative integers. Without loss of generality, we will assume that \( Y \) takes on the values 1, \ldots, \( k \). Each of the \( k \) outcomes has its own linear prediction. For the model to be identified, one of the outcomes is chosen to be the base or reference. The linear prediction for the base outcome is constrained to be 0 for all observations. Without loss of generality, we will assume the base outcome is the first outcome. Let \( z_i \) be the prediction for outcome \( i \), where \( z_1 = 0 \) for the base outcome.

Given the \( k \) linear predictions \( z' = (z_1, z_2, \ldots, z_k) \), the log of the conditional probability mass function is

\[
\log f(y | z) = z_y - \log \left\{ \sum_{i=1}^{k} \exp(z_i) \right\}
\]

The only link allowed for the multinomial family is the \texttt{logit} link.

**The Poisson family**

The Poisson family is a count-data response model. The response \( Y \) is assumed to take on nonnegative integer values.

The log of the conditional probability mass function is

\[
\log f(y | \mu) = -\mu + y \log \mu - \log \Gamma(y + 1)
\]

The only link allowed for the Poisson family is the \texttt{log} link.

**The negative binomial family**

The negative binomial family is another count-data response model. It is commonly thought of as a Poisson family with overdispersion. \texttt{gsem} allows two parameterizations for the dispersion in this family: mean dispersion and constant dispersion.

The log of the conditional probability mass function is

\[
\log f(y | \mu, \alpha) = \log \{ \Gamma(y + m) \} - \log \{ \Gamma(y + 1) \} - \log \{ \Gamma(m) \} \\
+ m \log p + y \log(1 - p)
\]

where \( m \) and \( p \) depend on the form of dispersion.
The only link allowed for the negative binomial family is the \textit{log} link.

For mean dispersion, we have

\[ m = \frac{1}{\alpha} \]
\[ p = \frac{1}{1 + \alpha \mu} \]

where \( \mu \) is the expected value of \( Y \) and \( \alpha \) is the scale parameter. \texttt{gsem} fits \( \alpha \) in the log scale.

For constant dispersion, we have

\[ m = \exp(\log \mu - \log \delta) \]
\[ p = \frac{1}{1 + \delta} \]

where \( \mu \) is the expected value of \( Y \) and \( \delta \) is the scale parameter. \texttt{gsem} fits \( \delta \) in the log scale.

\textbf{The Gaussian family}

The Gaussian family is a continuous response model and is synonymous with the normal distribution.

When the Gaussian family is specified with the \texttt{identity} link but no censoring, \texttt{gsem} fits this family by using a single multivariate density function and allows the following two special features:

1. \texttt{gsem} can fit covariances between the Gaussian error variables.
2. \texttt{gsem} can fit paths between Gaussian responses, including nonrecursive systems.

The log of the conditional probability density function is

\[ \log f(y|\mu, \Sigma) = -\frac{1}{2} \left\{ d \log 2\pi + \log |\Sigma| + (y - \mu)' \Sigma^{-1} (y - \mu) \right\} \]

where \( d \) is the dimension of the observed response vector \( y \), \( \mu \) is the mean of the responses, and \( \Sigma \) is the variance matrix of their unexplained errors.

When the Gaussian family is specified with the \texttt{log} link or censoring, the two special features described above no longer apply. In addition, the multivariate density function is no longer used. Instead, for each response using the \texttt{log} link, the log of the conditional probability density function corresponds to the formula above with \( d = 1 \). For censored responses, the log likelihood corresponds to the one in the \textit{Methods and formulas} for \texttt{R intreg}.

\textbf{Reliability}

For a given Gaussian response variable with the identity link, the reliability \( Y \) may be specified as \( p \) or \( 100 \times p \% \). The variance of \( Y \)'s associated error variable is then constrained to \((1 - p)\) times the observed variance of \( Y \).
Point mass

gSEM, with categorical latent variables, allows an outcome to have a unit probability mass at a single integer value. This feature is useful for fitting models like the zero-inflated Poisson model.

Link functions

Except for the ordinal and multinomial families, the link function defines the transformation between the mean and the linear prediction for a given response. If $Y$ is the random variable corresponding to an observed response variable $y$, then the link function performs the transformation

$$g(\mu) = z$$

where $\mu = E(Y)$ and $z$ is the linear prediction. In practice, the likelihood evaluator function uses the inverse of the link function to map the linear prediction to the mean.

The logit link

The logit link is

$$g(\mu) = \log(\mu) - \log(1 - \mu)$$

and its inverse is

$$\mu = g^{-1}(z) = \frac{1}{1 + e^z}$$

The probit link

The probit link is

$$g(\mu) = \Phi^{-1}(\mu)$$

and its inverse is

$$\mu = g^{-1}(z) = \Phi(z)$$

where $\Phi(\cdot)$ is the cumulative distribution function for the standard normal distribution and $\Phi^{-1}(\cdot)$ is its inverse.

The complementary log-log link

The complementary log-log link is

$$g(\mu) = \log\{-\log(1 - \mu)\}$$

and its inverse is

$$\mu = g^{-1}(z) = 1 - \exp\{-\exp(z)\}$$
The log link

The log link is

\[ g(\mu) = \log \mu \]

and its inverse is

\[ \mu = g^{-1}(z) = e^z \]

The identity link

The identity link is \( g(\mu) = \mu \).

Survival distributions

gsem also implements the most commonly used parametric distributions associated with survival analysis. This includes support for time-varying covariates and right-censoring. See [ST] streg for more background information on parametric survival models.

In this section, the observed endogenous variables are survival times. The random variable corresponding to a given response will continue to be denoted by \( Y \), with observed value denoted by \( y \) and expected value denoted by \( \mu \). The response \( Y \) is assumed to be a nonnegative real value. The starting time under observation is denoted by \( y_0 \), where \( y_0 \geq 0 \). The indicator for failure is denoted by \( d \), where \( d = 0 \) indicates a right-censored survival time, and \( d = 1 \) indicates failure at observed time \( y \).

The log likelihood for these models is given by

\[ l(y, y_0, d) = d \log f(y) + (1 - d) \log S(y) - \log S(y_0) \]

where \( S(\cdot) \) is the survivor function for \( Y \), and \( f(\cdot) \) is the corresponding density function. The relationship between the survivor function and the density is

\[ f(y) = -\frac{\partial S(y)}{\partial y} \]

In the following, we provide formulas for the survivor functions with respect to a linear prediction, denoted by \( z \).

The log link is the only link allowed with the survival distributions. Except for the gamma distribution, the use of the log link is merely a nod to the fact that the expected value of the outcome involves exponentiating the linear prediction. A formula for the expected value of \( Y \), as a function of the linear prediction, is provided for each survival distribution.
The exponential distribution

The survivor function for the exponential distribution is

$$S(y) = \exp(-\lambda y)$$

where $\lambda$ is a function of $z$ and is determined by one of the two parameterizations implemented in \texttt{gsem}. The proportional hazards parameterization is

$$\lambda = \exp(z)$$

and the accelerated failure-time parameterization is

$$\lambda = \exp(-z)$$

The expected value of $Y$ is

$$\mu = \frac{1}{\lambda}$$

The Weibull distribution

The survivor function for the Weibull distribution is

$$S(y) = \exp(-\lambda y^s)$$

where $s$ is a scale parameter, and $\lambda$ is a function of $z$ that is determined by one of the two parameterizations implemented in \texttt{gsem}. \texttt{gsem} fits $s$ in the log scale. The proportional hazards parameterization is

$$\lambda = \exp(z)$$

with expected value

$$\mu = \Gamma(1 + 1/s) \exp(-z/s)$$

and the accelerated failure-time parameterization is

$$\lambda = \exp(-sz)$$

with expected value

$$\mu = \Gamma(1 + 1/s) \exp(z)$$
The gamma distribution

The survivor function for the gamma distribution is

\[ S(y) = \int_y^\infty \frac{1}{\Gamma(s^{-2})} \left\{ e^{s^{-2}t} \right\}^{s^{-2}-1} \exp \left( -\frac{t}{e^{s^{-2}}} \right) \, dt \]

where \( s \) is a scale parameter. \texttt{gsem} fits \( s \) in the log scale.

The expected value for this distribution is

\[ \mu = \exp(z) \]

The loglogistic distribution

The survivor function for the loglogistic distribution is

\[ S(y) = \left\{ 1 + \exp \left( \frac{\log y - z}{s} \right) \right\}^{-1} \]

where \( s \) is a scale parameter. \texttt{gsem} fits \( s \) in the log scale.

The expected value for this distribution is

\[ \mu = \frac{\pi s \exp(z)}{\sin(\pi s)} \]

The lognormal distribution

The survivor function for the lognormal distribution is

\[ S(y) = 1 - \Phi \left( \frac{\log y - z}{s} \right) \]

where \( s \) is a scale parameter. \texttt{gsem} fits \( s \) in the log scale.

The expected value for this distribution is

\[ \mu = \exp \left( z + \frac{1}{2}s^2 \right) \]
Models with continuous latent variables

The likelihood

\texttt{gsem} fits generalized linear models with continuous latent variables via maximum likelihood. The likelihood for the specified model is derived under the assumption that each response variable is independent and identically distributed across the estimation sample. The response variables are also assumed to be independent of each other. These assumptions are conditional on the latent variables and the observed exogenous variables.

The likelihood is computed by integrating out the latent variables. Let \( \theta \) be the vector of model parameters, \( y \) be the vector of observed response variables, \( x \) be the vector of observed exogenous variables, and \( u \) be the \( r \times 1 \) vector of latent variables. Then, the marginal likelihood looks something like

\[
L(\theta) = \int_{\mathbb{R}^r} f(y|x, u, \theta) \phi(u|\mu_u, \Sigma_u) \, du
\]

where \( \mathbb{R} \) denotes the set of values on the real line, \( \mathbb{R}^r \) is the analog in \( r \)-dimensional space, \( \theta \) is a vector of the unique model parameters, \( f(\cdot) \) is the conditional probability density function for the observed response variables, \( \phi(\cdot) \) is the multivariate normal density for \( u \), \( \mu_u \) is the expected value of \( u \), and \( \Sigma_u \) is the covariance matrix for \( u \). All auxiliary parameters are fit directly without any further parameterization, so we simply acknowledge that the auxiliary parameters are among the elements of \( \theta \).

The \( y \) variables are assumed to be independent, conditionally on \( x \) and \( u \), so \( f(\cdot) \) is the product of the individual conditional densities. One exception to this is when \( y \) contains two or more Gaussian response variables with the identity link, in which case the Gaussian responses are actually modeled using a multivariate normal density to allow for correlated errors and nonrecursive systems among Gaussian responses. This one exception does not change how the integral is numerically evaluated, so we make no effort to represent this distinction in the formulas.

For a single-level model with \( n \) response variables, the conditional joint density function for a given observation is

\[
f(y|x, u, \theta) = \prod_{i=1}^{n} f_i(y_i|x, u, \theta)
\]

For a two-level model, the likelihood is computed at the cluster level, so the conditional density is also a product of the observation-level density contributions within a given cluster,

\[
f(y|x, u, \theta) = \prod_{i=1}^{n} \prod_{j=1}^{t} f_i(y_{ij}|x_j, u, \theta)
\]

where \( t \) is the number of individuals within the cluster. This extends to more levels by expanding the products down to the observations nested within the hierarchical groups. Because the single-level model is a special case of a two-level model where all the groups have a single observation, we will now use the two-level notation and subscripts.

Except for the ordinal and multinomial families, we use the link function to map the conditional mean

\[
\mu_{ij} = E(y_{ij}|x_j, u)
\]
to the linear prediction
\[ z_{ij} = x_j' \beta_i + x_j' \Lambda_i u \]
where \( \beta_i \) is the vector of the fixed-effect coefficients and \( \Lambda_i \) is the matrix of the latent loadings for \( y_{ij} \). For notational convenience, we will overload the definitions of \( f(\cdot) \) and \( f_i(\cdot) \) so that they are functions of the responses and model parameters through the linear predictions \( z' = (z_1, \ldots, z_n) \). Thus \( f(y|x, u, \theta) \) is equivalently specified as \( f(y, z, \theta) \), and \( f_i(y_{ij}|x_j, u, \theta) \) is equivalently specified as \( f_i(y_{ij}, z_{ij}, \theta) \). In this new notation, the likelihood for a given cluster is
\[
\mathcal{L}(\theta) = \int_{\mathbb{R}^r} f(y, z, \theta) \phi(u|\mu_u, \Sigma_u) \partial u = \frac{1}{(2\pi)^{r/2} \sqrt{|\Sigma_u|}} \int_{\mathbb{R}^r} \exp \left\{ \log f(y, z, \theta) - \frac{1}{2} (u - \mu_u)' \Sigma_u^{-1} (u - \mu_u) \right\} \partial u \tag{2}
\]
\texttt{gsem} allows nonrecursive systems between Gaussian response variables with the identity link, but non-Gaussian responses and Gaussian responses with the log link are not allowed to participate in any nonrecursive systems. This means that if a given response \( y \) is specified with a family other than Gaussian or a link other than identity, then \( y \) cannot have a path that ultimately leads back to itself. Any response may participate in a recursive system because the participating responses may be treated as exogenous variables when predicting other responses in a recursive system.

The latent vector \( u \) consists of stacked collections of the latent variables from each level. Within each level, the latent endogenous variables \( \eta \) are stacked over the latent exogenous variables \( \xi \). Within a given level, the latent exogenous variables and latent endogenous errors are assumed independent and multivariate normal,
\[
\xi \sim N(\kappa, \Phi) \\
\epsilon \sim N(0, \Psi)
\]
so according to the linear relationship
\[
\eta = B\eta + \Gamma \xi + A x + \epsilon
\]
we have that the latent variables are jointly multivariate normal. This linear relationship implies that \texttt{gsem} allows latent variables to predict each other but only within level. It also means that \texttt{gsem} allows paths from observed variables to latent variables; however, the observed variable must be constant within group if the path is to a group-level latent variable.

For our two-level model, we have
\[
u \sim N(\mu_u, \Sigma_u)
\]
where
\[
\mu_u = \begin{pmatrix} \mu_\eta \\ \kappa \end{pmatrix} \\
\Sigma_u = \begin{pmatrix} \Sigma_{\eta\eta} & \Sigma_{\eta\xi} \\ \Sigma_{\xi\eta} & \Phi \end{pmatrix}
\]
\[
\mu_\eta = (I - B)^{-1}(\Gamma \kappa + A x) \\
\Sigma_{\eta\eta} = (I - B)^{-1}(\Gamma \Phi \Gamma' + \Psi) \{(I - B)^{-1}\}' \\
\Sigma_{\eta\xi} = (I - B)^{-1}\Gamma \Phi
\]
The vector $\theta$ is therefore the set of unique model parameters taken from the following:

- $\beta_i$ is the vector of fixed-effect coefficients for $y_{ij}$.
- $\Lambda_i$ is the matrix of latent loadings for $y_{ij}$.
- $B$ is the matrix of latent endogenous coefficients.
- $\Gamma$ is the matrix of latent exogenous coefficients.
- $A$ is the matrix of latent fixed-effect coefficients.
- $\kappa$ is the vector of latent exogenous means.
- $\Phi$ is the matrix of latent exogenous variances and covariances.
- $\Psi$ is the matrix of latent endogenous error variances and covariances.

Auxiliary parameters are the result from some of the distribution families.

Each level of a multilevel model will have its own set of the following parameters: $B$, $\Gamma$, $A$, $\kappa$, $\Phi$, and $\Psi$. For multilevel models, $\Sigma_u$ is a block-diagonal matrix with a block for each level.

**Gauss–Hermite quadrature**

The integral in (2) is generally not tractable, so we must use numerical methods. In the univariate case, the integral of a function multiplied by the kernel of the standard normal distribution can be approximated using Gauss–Hermite quadrature (GHQ). For $q$-point GHQ, let the abscissa and weight pairs be denoted by $(a_k^*, w_k^*)$, $k = 1, \ldots, q$. The GHQ approximation is then

$$\int_{-\infty}^{\infty} f(x) \exp(-x^2) \, dx \approx \sum_{k=1}^{q} w_k^* f(a_k^*)$$

Using the standard normal distribution yields the approximation

$$\int_{-\infty}^{\infty} f(x)\phi(x) \, dx \approx \sum_{k=1}^{q} w_k f(a_k)$$

where $a_k = \sqrt{2}a_k^*$ and $w_k = (w_k^*)/\sqrt{\pi}$.

We can use a change-of-variables technique to transform the multivariate integral (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using GHQ. Let $v$ be a random vector whose elements are independently standard normal, and let $L$ be the Cholesky decomposition of $\Sigma_u$; that is, $\Sigma_u = LL'$. In the distribution, we have that $u = \mu_u + Lv$, and the linear predictions vector as a function of $v$ is

$$z_{ij} = x_j' \beta_i + x_j' \Lambda_i (\mu_u + Lv)$$

so the likelihood for a given cluster is

$$\mathcal{L}(\theta) = (2\pi)^{-r/2} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \exp \left\{ \log f(y, z, \theta) - \frac{1}{2} \sum_{k=1}^{r} v_k^2 \right\} \, dv_1 \cdots dv_r$$

where $r$ is the number of latent variables.
Consider an \( r \)-dimensional quadrature grid containing \( q \) quadrature points in each dimension. Let the vector of abscissas \( \mathbf{a}_k = (a_{k1}, \ldots, a_{kr})' \) be a point in this grid, and let \( \mathbf{w}_k = (w_{k1}, \ldots, w_{kr})' \) be the vector of corresponding weights. The GHQ approximation to the likelihood for a given cluster is

\[
L_{\text{GHQ}}(\theta) = \sum_{k_1=1}^{q} \cdots \sum_{k_r=1}^{q} \left[ \exp \left( \sum_{i=1}^{n} \log f_i(y_{ij}, z_{ijk}, \theta) \right) \right] \prod_{s=1}^{r} w_{ks}
\]

where

\[
z_{ijk} = x_j' \beta + x_j' \Lambda_i (\mu_u + L \alpha_k)
\]

### Adaptive quadrature

This section sets the stage for mean–variance adaptive Gauss–Hermite quadrature (MVAGH) and mode-curvature adaptive Gauss–Hermite quadrature (MCAGH).

Let’s reconsider the likelihood in (3). If we fix the observed variables and the model parameters, we see that the posterior density for \( v \) is proportional to

\[
\phi(v) f(y, z, \theta)
\]

It is reasonable to assume that this posterior density can be approximated by a multivariate normal density with mean vector \( \mu_v \) and variance matrix \( \tau_v \). Instead of using the prior density of \( v \) as the weighting distribution in the integral, we can use our approximation for the posterior density,

\[
L(\theta) = \int_{\mathbb{R}^r} \frac{f(y, z, \theta) \phi(v)}{\phi(v, \mu_v, \tau_v)} \phi(v, \mu_v, \tau_v) dv
\]

The likelihood is then approximated with

\[
L^*(\theta) = \sum_{k_1=1}^{q} \cdots \sum_{k_r=1}^{q} \left[ \exp \left( \sum_{i=1}^{n} \log f_i(y_{ij}, z_{ijk}^*, \theta) \right) \right] \prod_{s=1}^{r} \omega_{ks}
\]

where

\[
z_{ijk}^* = x_j' \beta + x_j' \Lambda_i (\mu_u + L \alpha_k)
\]

and \( \alpha_k \) and the \( \omega_{ks} \) are the adaptive versions of the abscissas and weights after an orthogonalizing transformation, which eliminates posterior covariances between the latent variables. \( \alpha_k \) and the \( \omega_{ks} \) are functions of \( \mathbf{a}_k \) and \( \mathbf{w}_k \) and the adaptive parameters \( \mu_v \) and \( \tau_v \).

For MVAGH, \( \mu_v \) is the posterior mean and \( \tau_v \) is the posterior variance of \( v \). They are computed iteratively by updating the posterior moments by using the MVAGH approximation, starting with a 0 mean vector and identity variance matrix.

For MCAGH, \( \mu_v \) is the posterior mode for \( v \), and \( \tau_v \) is the curvature at the mode. They are computed by optimizing the integrand in (3) with respect to \( v \).
Laplacian approximation

Let’s reconsider the likelihood in (2) and denote the argument in the exponential function by

\[ h(u) = \log f(y, z, \theta) - \frac{1}{2}(u - \mu_u)' \Sigma_u^{-1}(u - \mu_u) \]

\[ = \sum_{i=1}^n \sum_{j=1}^t \log f_i(y_{ij}, z_{ij}, \theta) - \frac{1}{2}(u - \mu_u)' \Sigma_u^{-1}(u - \mu_u) \]

where

\[ z_{ij} = x_j' \beta_i + x_j' \Lambda_i u \]

The Laplacian approximation is based on a second-order Taylor expansion of \( h(u) \) about the value of \( u \) that maximizes it. The first and second partials with respect to \( u \) are

\[ h'(u) = \frac{\partial h(u)}{\partial u} = \sum_{i=1}^n \sum_{j=1}^t \frac{\partial \log f_i(y_{ij}, z_{ij}, \theta)}{\partial z_{ij}} \Lambda_i x_j - \Sigma^{-1}(u - \mu_u) \]

\[ H(u) = \frac{\partial^2 h(u)}{\partial u \partial u'} = \sum_{i=1}^n \sum_{j=1}^t x_j' \Lambda_i \frac{\partial^2 \log f_i(y_{ij}, z_{ij}, \theta)}{\partial z_{ij} \partial z_{ij}} \Lambda_i x_j - \Sigma^{-1} \]

The maximizer of \( h(u) \) is \( \hat{u} \) such that \( h'(\hat{u}) = 0 \). The integral in (2) is proportional to the posterior density of \( u \) given the data, so \( \hat{u} \) is also the posterior mode.

The second-order Taylor approximation then takes the form

\[ h(u) \approx h(\hat{u}) + \frac{1}{2}(u - \hat{u})' H(\hat{u})(u - \hat{u}) \]

because the first-order derivative term is 0. The integral is approximated by

\[ \int_{\mathbb{R}^r} \exp\{h(u)\} \, du \approx \exp\{h(\hat{u})\} (2\pi)^{r/2} |-H(\hat{u})|^{-1/2} \]

because the second term in (4) is the kernel of a multivariate normal density once it is exponentiated. The Laplacian approximation for the log likelihood is

\[ \log L^{\text{Lap}}(\theta) = -\frac{1}{2} \log |\Sigma_u| - \frac{1}{2} \log |-H(\hat{u})| + h(\hat{u}) \]
Survey data

In the presence of sampling weights, following Rabe-Hesketh and Skrondal (2006), the weighted log pseudolikelihood for a two-level model is given as

$$\log L(\theta) = \sum_{j=1}^{M} w_j \log \int_{-\infty}^{\infty} \exp \left\{ \sum_{i=1}^{n_j} w_{i|j} \log f(y_{ij}|\eta_{ij}) \right\} \phi(v_{j1}) dv_{j1}$$

where $w_j$ is the inverse of the probability of selection for the $j$th cluster; $w_{i|j}$ is the inverse of the conditional probability of selection of individual $i$, given the selection of cluster $j$; $f(\cdot)$ is as defined previously; and $\phi(\cdot)$ is the standard multivariate normal density.

Weighted estimation is achieved through the direct application of $w_j$ and $w_{i|j}$ into the likelihood calculations as detailed above to reflect replicated clusters for $w_j$ and replicated observations within clusters for $w_{i|j}$. Because this estimation is based on replicated clusters and observations, frequency weights are handled similarly.

Weights are not allowed with crossed models or the Laplacian approximation.

Predictions

We begin by considering the prediction of the latent variables $u$ for a given cluster in a two-level model. Prediction of latent variables in multilevel generalized linear models involves assigning values to the latent variables, and there are many methods for doing so; see Skrondal and Rabe-Hesketh (2009) and Skrondal and Rabe-Hesketh (2004, chap. 7) for a comprehensive review. Stata offers two methods of predicting latent variables: empirical Bayes means (also known as posterior means) and empirical Bayes modes (also known as posterior modes). Below we provide more details about the two methods.

Empirical Bayes

Let $\hat{\theta}$ denote the estimated model parameters. Empirical Bayes (EB) predictors of the latent variables are the means or modes of the empirical posterior distribution with the parameter estimates $\theta$ replaced with their estimates $\hat{\theta}$. The method is called “empirical” because $\hat{\theta}$ is treated as known. EB combines the prior information about the latent variables with the likelihood to obtain the conditional posterior distribution of latent variables. Using Bayes’s theorem, the empirical conditional posterior distribution of the latent variables for a given cluster is

$$\omega(u|y, x; \hat{\theta}) = \frac{f(y|u, x; \hat{\theta}) \phi(u; \hat{\mu}_u, \hat{\Sigma}_u)}{\int f(y|u, x; \hat{\theta}) \phi(u; \hat{\mu}_u, \hat{\Sigma}_u) du}$$

The denominator is just the likelihood contribution of the given cluster.

EB mean predictions of latent variables, $\tilde{u}$, also known as posterior means, are calculated as

$$\tilde{u} = \int_{\mathbb{R}^c} u \omega(u|y, x; \hat{\theta}) du$$
where we use the notation $\tilde{u}$ rather than $\hat{u}$ to distinguish predicted values from estimates. This multivariate integral is approximated by MVAGH. If you have multiple latent variables within a level or latent variables across levels, the calculation involves orthogonalizing transformations with the Cholesky transformation because the latent variables are no longer independent under the posterior distribution.

When all the response variables are normal, the posterior density is multivariate normal, and EB means are also best linear unbiased predictors (BLUPs); see Skrondal and Rabe-Hesketh (2004, 227). In generalized mixed-effects models, the posterior density tends to multivariate normal as cluster size increases.

EB modal predictions can be approximated by solving for $\tilde{u}$ such that

$$\frac{\partial}{\partial u} \log \omega(u|y, x; \hat{\theta}) \bigg|_{u=\tilde{u}} = 0$$

Because the denominator in $\omega(\cdot)$ does not depend on $u$, we can omit it from the calculation to obtain the EB mode. The calculation of EB modes does not require numerical integration; thus they are often used in place of EB means. As the posterior density gets closer to being multivariate normal, EB modes get closer and closer to EB means.

Just like there are many methods of assigning values to the random effects, there are many methods of calculating standard errors of the predicted random effects; see Skrondal and Rabe-Hesketh (2009) for a comprehensive review.

Stata uses the posterior standard deviation as the standard error of the posterior means predictor of random effects. For a given level, the EB posterior covariance matrix of the random effects is given by

$$\text{Cov}(\tilde{u}|y, x; \hat{\theta}) = \int_{\mathbb{R}^r} (u - \tilde{u})(u - \tilde{u})' \omega(u|y, x; \hat{\theta}) \, du$$

The posterior covariance matrix and the integrals are approximated by MVAGH.

Conditional standard errors for the estimated posterior modes are derived from standard theory of maximum likelihood, which dictates that the asymptotic variance matrix of $\tilde{u}$ is the negative inverse of the Hessian matrix.

**Other predictions**

In what follows, we show formulas with the posterior means estimates of random effects $\tilde{u}$, which are used by default or if the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{u}$ are simply replaced with the posterior modes $\tilde{\tilde{u}}$ in these formulas.

For the $i$th response in the $j$th observation within a given cluster in a two-level model, the linear predictor is computed as

$$z_{ij} = x_j' \hat{\beta} + x_j' \Lambda_i \tilde{u}$$

The linear predictor includes the offset or exposure variable if one was specified during estimation, unless the `nooffset` option is specified. If the `fixedonly` option is specified, the linear predictor is computed as

$$\hat{z}_{ij} = x_j' \hat{\beta}$$
The predicted mean, conditional on the predicted latent variables, is

\[ \hat{\mu}_{ij} = g^{-1}(\hat{z}_{ij}) \]

where \( g^{-1}(\cdot) \) is the inverse link function defined in *Link functions* above. For the ordinal and multinomial families, the predicted mean is actually a probability, and \texttt{gsem} can produce a probability for each outcome value as described in *The ordinal family* and *The multinomial family* above. For the survival distributions, the predicted mean is computed using the formulas provided in *Survival distributions* above.

**Models with categorical latent variables**

**The likelihood**

\texttt{gsem} fits generalized linear models with categorical latent variables via maximum likelihood. The likelihood for the specified model is derived under the assumption that each response variable is independent and identically distributed across the estimation sample. The response variables are also assumed to be independent of each other. These assumptions are conditional on the categorical latent variables and the observed exogenous variables.

The likelihood is computed by combining the conditional likelihoods from each latent class (level combinations of the categorical latent variables) weighted by the associated latent class probabilities. Let \( \theta \) be the vector of model parameters. For a given observation, let \( y \) be the vector of observed response variables and \( x \) be the vector of observed exogenous variables. Without loss of generality, we will assume a single categorical latent variable \( C \) with \( k \) levels \( 1, \ldots, k \). The marginal likelihood for a given observation looks something like

\[ L_C(\theta) = \sum_{i=1}^{k} \pi_i f_i(y|x, c_i = 1, \theta) \]

where \( \pi_i \) is the probability for the \( i \)th latent class, \( f_i(\cdot) \) is the conditional probability density function for the observed response variables in the \( i \)th latent class, and \( c' = (c_1, \ldots, c_k) \) is the vector of latent class indicators. When \( c_i = 1 \), all other elements of \( c \) are 0. All auxiliary parameters are fit directly without any further parameterization, so we simply acknowledge that the auxiliary parameters are among the elements of \( \theta \).

The \( y \) variables are assumed to be independent, conditionally on \( x \) and \( C \), so \( f_i(\cdot) \) is the product of the individual conditional densities. One exception to this is when \( y \) contains two or more Gaussian response variables with the identity link, in which case the Gaussian responses are actually modeled using a multivariate normal density to allow for correlated errors and nonrecursive systems among Gaussian responses. This one exception does not meaningfully change the following discussion, so we make no effort to represent this distinction in the formulas.

For the \( i \)th latent class with \( n \) response variables, the conditional joint density function for a given observation is

\[ f_i(y|x, \theta) = \prod_{j=1}^{n} f_{ij}(y_{ij}|x, \theta) \]
Except for the ordinal and multinomial families, we use the link function to map the conditional mean

$$\mu_{ij} = E(y_{ij} | x, c_i = 1)$$

to the linear prediction

$$z_{ij} = x' \beta_{ij}$$

where $\beta_{ij}$ is the vector of the coefficients for $y_{ij}$. For notational convenience, we will overload the definitions of $f_i(\cdot)$ and $f_{ij}(\cdot)$ so that they are functions of the responses and model parameters through the linear predictions $z'_i = (z_{i1}, \ldots, z_{in})$. Thus $f_i(y | x, \theta)$ is equivalently specified as $f_i(y, z_i, \theta)$, and $f_{ij}(y_{ij} | x, \theta)$ is equivalently specified as $f_{ij}(y_{ij}, z_{ij}, \theta)$. In this new notation, the likelihood for a given observation is

$$L(\theta) = \sum_{i=1}^{k} \pi_i \prod_{j=1}^{n} f_{ij}(y_{ij}, z_{ij}, \theta)$$

\text{gsem} uses the multinomial distribution to model the probabilities for the latent classes. For the $i$th latent class, the probability is given by

$$\pi_i = \Pr(c_i = 1 | x) = \frac{\exp(z_i)}{\sum_a \exp(z_a)}$$

where the linear prediction for the $i$th latent class is

$$z_i = x' \gamma_i$$

and $\gamma_i$ is the associated vector of coefficients. \text{gsem} assumes the first latent class is the base level; $\gamma_1$ is a vector of zeros so that $z_1 = 0$ and $\exp(z_1) = 1$.

The vector $\theta$ is therefore the set of unique model parameters taken from the following:

$\gamma_i$ is the vector of coefficients for the $i$th latent class.

$\beta_{ij}$ is the vector of coefficients for $y_{ij}$.

Auxiliary parameters that result from some of the distribution families.

Each latent class will have its own set of these parameters.

The EM algorithm

\text{gsem} uses the EM algorithm to refine starting values before maximizing the likelihood in (1).

The EM algorithm uses the complete-data likelihood, a likelihood where it is as if we have observed values for the latent class indicator variables $c$. In the complete-data case, the likelihood for a given observation is

$$L(\theta) = \prod_{i=1}^{k} \{\pi_i f_i(y, z_i, \theta)\}^{c_i}$$
so the complete-data log likelihood is

$$\log L(\theta) = \sum_{i=1}^{k} c_i \{ \log \pi_i + \log f_i(y, z_i, \theta) \}$$

We intend to maximize the expected complete-data log likelihood given the observed variables $y$ and $x$. This is an iterative process in which we use the $g$th guess of the model parameters, denoted $\theta_{(g)}$, then compute the next guess $\theta_{(g+1)}$.

In the expectation (E) step, we derive the functional form of the expected complete-data log likelihood. The complete-data log likelihood is a linear function of the latent class indicator variables, so

$$E(c_i|y, x, \theta_{(g)}) = \frac{\pi_i f_i(y, z_i, \theta_{(g)})}{\sum_{j=1}^{k} \pi_j f_j(y, z_j, \theta_{(g)})}$$

We denote this posterior probability by $p_i$, so the expected complete-data log likelihood for a given observation is given by

$$Q(\theta|\theta_{(g)}) = \sum_{i=1}^{k} p_i \{ \log \pi_i + \log f_i(y, z_i, \theta) \}$$

Note that $Q(\theta|\theta_{(g)})$ is a function of $\theta_{(g)}$ solely through the posterior probabilities $p_i$.

Now that we have the conditional complete-data log likelihood, the maximization (M) step is to maximize $Q(\theta|\theta_{(g)})$ with respect to $\theta$ to find $\theta_{(g+1)}$.

**Survey data**

gsem with categorical latent variables also supports estimation with survey data; however, only the linearized variance estimator is supported. For details on VCEs with survey data, see [SVY] Variance estimation.

**Predictions**

The predicted mean for a given response within a latent class is computed by applying the associated link function to the linear prediction; see Link functions above. For ordinal and multinomial responses, the predicted mean for a given response level is the predicted probability for that level. For survival outcomes, the formulas for predicted means are provided in Survival distributions above.

Let $\hat{z}_i$ be the linear prediction for the $i$th latent class; then, the predicted probability for the $i$th latent class is given by

$$\hat{\pi}_i = \frac{\exp(\hat{z}_i)}{\sum_{a=1}^{k} \exp(\hat{z}_a)}$$
The predicted posterior probability for the $i$th latent class is given by

$$
\tilde{\pi}_i = \frac{\hat{\pi}_i f_i(y, \hat{z}_i, \hat{\theta})}{\sum_{j=1}^{k} \hat{\pi}_j f_j(y, \hat{z}_j, \hat{\theta})}
$$

Let $\hat{\mu}_i$ be the predicted mean of response $y$ in the $i$th latent class. The predicted overall mean of $y$, using the fitted latent class probabilities, is given by

$$
\hat{\mu} = \sum_{i=1}^{k} \hat{\pi}_i \hat{\mu}_i
$$

The predicted overall mean of $y$, using the posterior latent class probabilities, is given by

$$
\tilde{\mu} = \sum_{i=1}^{k} \tilde{\pi}_i \hat{\mu}_i
$$

References


Also see

[SEM] gsem — Generalized structural equation model estimation command
Description

The methods and formulas for the sem commands are presented below.

Remarks and examples

Remarks are presented under the following headings:

- Variable notation
- Model and parameterization
- Summary data
- Maximum likelihood
- Weighted least squares
- Groups
- Fitted parameters
- Satorra–Bentler variance estimation
- Standardized parameters
- Reliability
- Postestimation
  - Model framework
  - Goodness of fit
  - Group goodness of fit
  - Equation-level goodness of fit
  - Wald tests
  - Score tests
  - Residuals
  - Testing standardized parameters
  - Stability of nonrecursive systems
  - Direct, indirect, and total effects
  - Predictions

Variable notation

We will use the following convention to keep track of the five variable types recognized by the sem estimation command:

1. Observed endogenous variables are denoted $y$.
2. Observed exogenous variables are denoted $x$.
3. Latent endogenous variables are denoted $\eta$.
4. Latent exogenous variables are denoted $\xi$.
5. Error variables are denoted with prefix $e$. on the associated endogenous variable.
   a. Error variables for observed endogenous are denoted $e.y$.
   b. Error variables for latent endogenous are denoted $e.\eta$. 
In any given analysis, there are typically several variables of each type. Vectors of the four main variable types are denoted $y$, $x$, $\eta$, and $\xi$. The vector of all endogenous variables is

$$Y = \begin{pmatrix} y \\ \eta \end{pmatrix}$$

The vector of all exogenous variables is

$$X = \begin{pmatrix} x \\ \xi \end{pmatrix}$$

The vector of all error variables is

$$\zeta = \begin{pmatrix} e.y \\ e.\eta \end{pmatrix}$$

**Model and parameterization**

sem can fit models of the form

$$Y = BY + GX + \alpha + \zeta$$

where $B = [\beta_{ij}]$ is the matrix of coefficients on endogenous variables predicting other endogenous variables, $G = [\gamma_{ij}]$ is the matrix of coefficients on exogenous variables, $\alpha = [\alpha_i]$ is the vector of intercepts for the endogenous variables, and $\zeta$ is assumed to have mean 0 and

$$\text{Cov}(X, \zeta) = 0$$

Let

$$\kappa = [\kappa_j] = E(X)$$

$$\Phi = [\phi_{ij}] = \text{Var}(X)$$

$$\Psi = [\psi_{ij}] = \text{Var}(\zeta)$$

Then the mean vector of the endogenous variables is

$$\mu_Y = E(Y) = (I - B)^{-1}(G\kappa + \alpha)$$

the variance matrix of the endogenous variables is

$$\Sigma_{YY} = \text{Var}(Y) = (I - B)^{-1}(G\Phi G' + \Psi) \{ (I - B)^{-1} \}'$$

and the covariance matrix between the endogenous variables and the exogenous variables is

$$\Sigma_{YX} = \text{Cov}(Y, X) = (I - B)^{-1} G \Phi$$

Let $Z$ be the vector of all variables:

$$Z = \begin{pmatrix} Y \\ X \end{pmatrix}$$

Then its mean vector is

$$\mu = E(Z) = \begin{pmatrix} \mu_Y \\ \kappa \end{pmatrix}$$

and its variance matrix is

$$\Sigma = \text{Var}(Z) = \begin{pmatrix} \Sigma_{YY} & \Sigma_{YX} \\ \Sigma_{X}' & \Phi \end{pmatrix}$$
Summary data

Let \( z_t \) be the vector of all observed variables for the \( t \)th observation,
\[
  z_t = \begin{pmatrix} y_t \\ x_t \end{pmatrix}
\]
and let \( w_t \) be the corresponding weight value, where \( t = 1, \ldots, N \). If no weights were specified, then \( w_t = 1 \). Let \( w \) be the sum of the weights; then the sample mean vector is
\[
  \bar{z} = \frac{1}{w} \sum_{t=1}^{N} w_t z_t
\]
and the sample variance matrix is
\[
  S = \frac{1}{w - 1} \sum_{t=1}^{N} w_t (z_t - \bar{z})(z_t - \bar{z})'
\]

Maximum likelihood

Let \( \theta \) be the vector of unique model parameters, such as
\[
  \theta = \begin{pmatrix} \text{vec}(B) \\ \text{vec}(\Gamma) \\ \text{vech}(\Psi) \\ \text{vech}(\Phi) \\ \alpha \\ \kappa \end{pmatrix}
\]
Then under the assumption of the multivariate normal distribution, the overall log likelihood for \( \theta \) is
\[
  \log L(\theta) = -\frac{w}{2} \left[ k \log(2\pi) + \log \left\{ \det(\Sigma_o) \right\} + \text{tr} \left( D\Sigma_o^{-1} \right) \right]
\]
where \( k \) is the number of observed variables, \( \Sigma_o \) is the submatrix of \( \Sigma \) corresponding to the observed variables, and
\[
  D = fS + (\bar{z} - \mu_o)(\bar{z} - \mu_o)'
\]
where
\[
  f = \begin{cases} 
  1, & \text{if\ nm1 is specified} \\
  \frac{w - 1}{w}, & \text{otherwise}
\end{cases}
\]
and \( \mu_o \) is the subvector of \( \mu \) corresponding to the observed variables.

For the BHHH optimization technique and when computing observation-level scores, the log likelihood for \( \theta \) is computed as
\[
  \log L(\theta) = -\sum_{t=1}^{N} \frac{w_t}{2} \left[ k \log(2\pi) + \log \left\{ \det(\Sigma_o) \right\} + (z_t - \mu_o)'\Sigma_o^{-1}(z_t - \mu_o) \right]
\]
and the nm1 option is ignored.

When method(mlmv) is specified, \texttt{sem} groups the data according to missing-value patterns. Each missing-value pattern will have its own summary data: \( k, \bar{z}, \text{and} \ S \). The log likelihood for a missing-value pattern is computed using this summary data and the corresponding elements of \( \mu_o \) and \( \Sigma_o \). The overall log likelihood is computed by summing the log-likelihood values from each missing-value pattern.
Weighted least squares

Let $v$ be the vector of unique sample moments of the observed variables and $\tau$ be the corresponding vector of population moments. Then

$$v = \begin{bmatrix} z_{veh(fS)} \end{bmatrix}$$

and

$$\tau = \begin{bmatrix} \mu_{o_{veh}(\Sigma_o)} \end{bmatrix}$$

The weighted least-squares (WLS) criterion function to minimize is the quadratic form

$$F_{WLS}(\theta) = (v - \tau)'W^{-1}(v - \tau)$$

where $W$ is the least-squares weight matrix. For unweighted least squares (ULS), the weight matrix is the identity matrix $W = I$. Other weight matrices are mentioned in Bollen (1989).

The weight matrix implemented in \texttt{sem} is an estimate of the asymptotic covariance matrix of $v$. This weight matrix is derived without any distributional assumptions and is often referred to as derived from an arbitrary distribution function or is asymptotic distribution free (ADF), thus the option \texttt{method(adf)}.

Groups

When the \texttt{group()} option is specified, each group has its own summary data and model parameters. The entire collection of model parameters is

$$\theta = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_G \end{pmatrix}$$

where $G$ is the number of groups. The group-level criterion values are combined to produce an overall criterion value.

For \texttt{method(ml)} and \texttt{method(mlmv)}, the overall log likelihood is

$$\log L(\theta) = \sum_{g=1}^{G} \log L(\theta_g)$$

For \texttt{method(adf)}, the overall criterion function to minimize is

$$F_{WLS}(\theta) = \sum_{g=1}^{G} F_{WLS}(\theta_g)$$
Fitted parameters

`sem` fits the specified model by maximizing the log likelihood or minimizing the WLS criterion. If \( \theta \) is the vector of model parameters, then the fitted parameter vector is denoted by \( \hat{\theta} \), and similarly for \( \hat{B}, \hat{\Gamma}, \hat{\Psi}, \hat{\Phi}, \hat{\alpha}, \hat{\kappa}, \hat{\Sigma}, \hat{\mu} \), and their individual elements.

Satorra–Bentler variance estimation

When `vce(sbentler)` is specified, the variance of \( \hat{\theta} \) is estimated as

\[
\hat{V}_{sb}(\hat{\theta}) = \frac{1}{w}(\Delta' H \Delta)^{-1} \Delta' W H W \Delta (\Delta' H \Delta)^{-1}
\]

where \( W \) is the asymptotic covariance matrix of \( v \) used in the WLS criterion function, and \( \Delta \) is the partial derivative matrix of \( \tau \) with respect to \( \theta \) and uses the maximum likelihood fitted values \( \hat{\theta} \). \( H \) is computed as

\[
H = \begin{pmatrix}
S^{-1} & 0 \\
0 & A
\end{pmatrix}
\]

where

\[
A = 0.5 D' (S^{-1} \otimes S^{-1}) D
\]

and \( D \) is a duplication matrix. See Satorra and Bentler (1994, eq. 16.10).

Standardized parameters

Let \( \hat{\sigma}_{ii} \) be the \( i \)th diagonal element of \( \hat{\Sigma}_{YY} \). Then the standardized parameter estimates are

\[
\tilde{\beta}_{ij} = \hat{\beta}_{ij} \sqrt{\frac{\hat{\sigma}_{ii}}{\hat{\sigma}_{jj}}}
\]

\[
\tilde{\gamma}_{ij} = \hat{\gamma}_{ij} \sqrt{\frac{\hat{\phi}_{ii}}{\hat{\sigma}_{jj}}}
\]

\[
\tilde{\psi}_{ij} = \begin{cases}
\hat{\psi}_{ij} / \sqrt{\hat{\psi}_{ii} \hat{\psi}_{jj}}, & \text{if } i \neq j \\
\hat{\psi}_{ii} / \sqrt{\hat{\sigma}_{ii}}, & \text{if } i = j
\end{cases}
\]

\[
\tilde{\phi}_{ij} = \frac{\hat{\phi}_{ij}}{\sqrt{\hat{\phi}_{ii} \hat{\phi}_{jj}}}
\]

\[
\tilde{\alpha}_i = \frac{\hat{\alpha}_i}{\sqrt{\hat{\sigma}_{ii}}}
\]

\[
\tilde{\kappa}_j = \frac{\hat{\kappa}_j}{\sqrt{\hat{\phi}_{jj}}}
\]

The variance matrix of the standardized parameters is estimated using the delta method.
Reliability

For an observed endogenous variable, \( y \), the reliability may be specified as \( p \) or 100 \( \times \) \( p\% \). The variance of \( e.y \) is then constrained to \((1 - p)\) times the observed variance of \( y \).

Postestimation

Model framework

`estat framework` reports the fitted parameters in their individual matrix forms as introduced in Fitted parameters.

Goodness of fit

`estat gof` reports the following goodness-of-fit statistics.

Let the degrees of freedom for the specified model be denoted by \( df_m \). In addition to the specified model, `sem` fits saturated and baseline models corresponding to the observed variables in the specified model. The saturated model fits a full covariance matrix for the observed variables and has degrees of freedom

\[
df_s = \left( \frac{p + q + 1}{2} \right) + p + q
\]

where \( p \) is the number of observed endogenous variables and \( q \) is the number of observed exogenous variables in the model. The baseline model fits a reduced covariance matrix for the observed variables depending on the presence of endogenous variables. If there are no endogenous variables, all variables are uncorrelated in the baseline model; otherwise, only exogenous variables are correlated in the baseline model. The degrees of freedom for the baseline model is

\[
df_b = \begin{cases} 
2q, & \text{if } p = 0 \\
2p + q + \left( \frac{q + 1}{2} \right), & \text{if } p > 0 
\end{cases}
\]

For `method(ml)` and `method(mlmv)`, let the saturated log likelihood be denoted by \( \log L_s \) and the baseline log likelihood be denoted by \( \log L_b \). The likelihood-ratio test of the baseline versus saturated models is computed as

\[
\chi^2_{bs} = 2(\log L_s - \log L_b)
\]

with degrees of freedom \( df_{bs} = df_s - df_b \). The likelihood-ratio test of the specified model versus the saturated model is computed as

\[
\chi^2_{ms} = 2\{ \log L_s - \log L(\hat{\theta}) \}
\]

with degrees of freedom \( df_{ms} = df_s - df_m \).
The Satorra–Bentler (1994) scaled \( \chi^2 \) test is computed as
\[
\chi^2_{sb,ms} = \chi^2_{ms}/c
\]
where \( c = \text{trace}(UW/df_{ms}) \). \( U \) is computed as
\[
U = H - H\Delta(\Delta' H\Delta)^{-1}\Delta' H
\]
where \( \Delta \) and \( H \) are defined as in Satorra–Bentler variance estimation above.

The Satorra–Bentler scaled \( \chi^2 \) test for the baseline versus saturated test, \( \chi^2_{sb,bs} \), is defined as above, but \( \Delta \) is replaced for the specified model with \( \Delta \) for the baseline model.

For `method(adf)` the saturated criterion value is 0, \( F_s = 0 \). Let the baseline criterion value be denoted by \( F_b \). The \( \chi^2 \) test of the baseline versus saturated models is computed as
\[
\chi^2_{bs} = N F_b
\]
with degrees of freedom \( df_{bs} = df_s - df_b \). The chi-squared test of the specified model versus the saturated model is computed as
\[
\chi^2_{ms} = N F_{WLS}(\hat{\theta})
\]
with degrees of freedom \( df_{ms} = df_s - df_m \).

The Akaike information criterion (Akaike 1974) is defined as
\[
\text{AIC} = -2 \log L(\hat{\theta}) + 2df_m
\]
The Bayesian information criterion (Schwarz 1978) is defined as
\[
\text{BIC} = -2 \log L(\hat{\theta}) + N df_m
\]
See [R] BIC note for additional information on calculating and interpreting the BIC.

The overall coefficient of determination is computed as
\[
CD = 1 - \frac{\text{det}(\hat{\Psi})}{\text{det}(\hat{\Sigma}_{YY})}
\]
This value is also referred to as the overall \( R^2 \) in `estat eqgof` (see [SEM] `estat eqgof`).

The root mean squared error of approximation (Browne and Cudeck 1993) is computed as
\[
\text{RMSEA} = \left\{ \frac{(\chi^2_{ms} - df_{ms})G}{(N-1)df_{ms}} \right\}^{1/2}
\]
The 90% confidence interval for RMSEA is
\[
\left( \sqrt{G\lambda_L/(N-1)df_{ms}}, \sqrt{G\lambda_U/(N-1)df_{ms}} \right)
\]
where \( \lambda_L \) and \( \lambda_U \) are the noncentrality parameters corresponding to a noncentral chi-squared distribution with \( df_{ms} \) degrees of freedom in which the noncentral chi-squared random variable has a cumulative distribution function equal to 0.95 and 0.05, respectively.
The Browne and Cudeck (1993) p-value for the test of close fit with null hypothesis

\[ H_0: \text{RMSEA} \leq 0.05 \]

is computed as

\[ p = 1 - \Pr(\chi^2 < \chi^2_{\text{ms}}|\lambda, \text{df}_{\text{ms}}) \]

where \( \chi^2 \) is distributed noncentral chi-squared with noncentrality parameter \( \lambda = (0.05)^2(N-1)\text{df}_{\text{ms}} \) and \( \text{df}_{\text{ms}} \) degrees of freedom. This p-value is not computed when the `group()` option is specified.

The comparative fit index (Bentler 1990) is computed as

\[
\text{CFI} = 1 - \left[ \frac{(\chi^2_{\text{ms}} - \text{df}_{\text{ms}})}{\max\{(\chi^2_{\text{bs}} - \text{df}_{\text{bs}}), (\chi^2_{\text{ms}} - \text{df}_{\text{ms}})\}} \right]
\]

The Tucker–Lewis index (Bentler 1990) is computed as

\[
\text{TLI} = \frac{(\chi^2_{\text{bs}}/\text{df}_{\text{bs}}) - (\chi^2_{\text{ms}}/\text{df}_{\text{ms}})}{(\chi^2_{\text{bs}}/\text{df}_{\text{bs}}) - 1}
\]

Let \( k \) be the number of observed variables in the model. The standardized root mean squared residual is computed as

\[
\text{SRMR} = \left\{ \frac{2 \sum_{i=1}^{k} \sum_{j<i} r_{ij}^2}{k(k+1)G} \right\}^{1/2}
\]

where \( r_{ij} \) is the standardized covariance residual

\[
r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}} = \frac{\hat{\sigma}_{ij}}{\sqrt{\hat{\sigma}_{ii}\hat{\sigma}_{jj}}}
\]

These standardized residuals are not the same as those reported by `estat residuals`; see `Residuals` below.

The RMSEA, CFI, and TLI are also reported using Satorra–Bentler scaled statistics when the model is fit using `vce(sbentler)`. In the formulas above, \( \chi^2_{\text{ms}} \) and \( \chi^2_{\text{bs}} \) are replaced, respectively, with \( \chi^2_{\text{sb,ms}} \) and \( \chi^2_{\text{sb,bs}} \).

**Group goodness of fit**

`estat ggof` reports CD, SRMR, and model versus saturated \( \chi^2 \) values for each group separately. The group-level formulas are the same as those computed for a single group analysis; see `Goodness of fit` above.

**Equation-level goodness of fit**

`estat eqgof` reports goodness-of-fit statistics for each endogenous variable in the specified model. The coefficient of determination for the \( i \)th endogenous variable is computed as

\[
R_i^2 = 1 - \frac{\hat{\psi}_{ii}}{\hat{\sigma}_{ii}}
\]
The Bentler–Raykov (Bentler and Raykov 2000) squared multiple correlation for the \( i \)th endogenous variable is computed as
\[
mc_i^2 = \frac{\hat{\text{Cov}}(y_i, \hat{y}_i)}{\sqrt{\hat{\sigma}_{ii} \hat{\text{Var}}(\hat{y}_i)}}
\]
where \( \hat{\sigma}_{ii} \) is a diagonal element of \( \hat{\Sigma} \), \( \hat{\text{Var}}(\hat{y}_i) \) is a diagonal element of
\[
\hat{\text{Var}}(\hat{Y}) = (I - \hat{B})^{-1} \hat{\Phi} \hat{\Gamma}' \left\{ (I - \hat{B})^{-1} - I \right\} \hat{\Psi} \left\{ (I - \hat{B})^{-1} - I \right\}'
\]
and \( \hat{\text{Cov}}(y_i, \hat{y}_i) \) is a diagonal element of
\[
\hat{\text{Cov}}(Y, \hat{Y}) = (I - \hat{B})^{-1} \hat{\Phi} \hat{\Gamma}' \left\{ (I - \hat{B})^{-1} \right\}' + (I - \hat{B})^{-1} \hat{\Psi} \left\{ (I - \hat{B})^{-1} - I \right\}'
\]

**Wald tests**

`estat eqtest` performs Wald tests on the coefficients for each endogenous equation in the model. `estat ginvariant` computes a Wald test of group invariance for each model parameter that is free to vary across all groups. See \([R]\) `test`.

**Score tests**

`estat mindices` computes modification indices for each constrained parameter in the model, including paths and covariances that were not even part of the model specification. Modification indices are score tests, which are also known as Lagrange multiplier tests. `estat scoretests` performs a score test for each user-specified linear constraint. `estat ginvariant` performs a score test of group invariance for each model parameter that is constrained to be equal across all groups.

A score test compares a constrained model fit to the same model without one or more constraints. The score test is computed as
\[
\chi^2 = g(\hat{\theta})'V(\hat{\theta})g(\hat{\theta})
\]
where \( \hat{\theta} \) is the fitted parameter vector from the constrained model, \( g(\theta) \) is the gradient vector function for the unconstrained model, and \( V(\theta) \) is the variance matrix function computed from the expected information matrix function for the unconstrained model. For `method(ml)` and `method(mlmv)`,
\[
g(\theta) = \frac{\partial \log L^*(\theta)}{\partial \theta}
\]
\[
V(\theta) = \left[ E \left\{ -\frac{\partial^2 \log L^*(\theta)}{\partial \theta \partial \theta'} \right\} \right]^{-1}
\]
where \( \log L^*(\theta) \) is the log-likelihood function for the unconstrained model. For `method(adf)`,
\[
g(\theta) = -\frac{\partial F^*_{WLS}(\theta)}{\partial \theta}
\]
\[
V(\theta) = \left[ E \left\{ \frac{\partial^2 F^*_{WLS}(\theta)}{\partial \theta \partial \theta'} \right\} \right]^{-1}
\]
where \( F^*(\theta) \) is the WLS criterion function for the unconstrained model.
The score test is computed as described in Wooldridge (2010) when \texttt{vce(robust)} or \texttt{vce(cluster clustvar)} is specified.

**Residuals**

\texttt{estat residuals} reports raw, normalized, and standardized residuals for means and covariances of the observed variables.

The raw residual for the mean of the \(i\)th observed variable is

\[ z_i - \hat{\mu}_i \]

The raw residual for the covariance between the \(i\)th and \(j\)th observed variables is

\[ S_{ij} - \hat{\sigma}_{ij} \]

The normalized residual for the mean of the \(i\)th observed variable is

\[ \frac{z_i - \hat{\mu}_i}{\sqrt{\hat{\text{Var}}(z_i)}} \]

where

\[ \hat{\text{Var}}(z_i) = \begin{cases} S_{ii}, & \text{if the sample option is specified} \\ \frac{\hat{\sigma}_{ii}}{N}, & \text{otherwise} \end{cases} \]

The normalized residual for the covariance between the \(i\)th and \(j\)th observed variables is

\[ \frac{S_{ij} - \hat{\sigma}_{ij}}{\sqrt{\hat{\text{Var}}(S_{ij})}} \]

where

\[ \hat{\text{Var}}(S_{ij}) = \begin{cases} \frac{S_{ii}S_{jj} + S_{ij}}{N}, & \text{if the sample option is specified} \\ \frac{\hat{\sigma}_{ii}\hat{\sigma}_{jj} + \hat{\sigma}_{ij}}{N}, & \text{otherwise} \end{cases} \]

If the \texttt{nm1} option is specified, the denominator in the variance estimates is \(N - 1\) instead of \(N\).

The standardized residual for the mean of the \(i\)th observed variable is

\[ \frac{z_i - \hat{\mu}_i}{\sqrt{\hat{\text{Var}}(z_i - \hat{\mu}_i)}} \]

where

\[ \hat{\text{Var}}(z_i - \hat{\mu}_i) = \hat{\text{Var}}(z_i) - \hat{\text{Var}}(\hat{\mu}_i) \]

and \(\hat{\text{Var}}(\hat{\mu}_i)\) is computed using the delta method. Missing values are reported when the computed value of \(\hat{\text{Var}}(z_i)\) is less than \(\hat{\text{Var}}(\hat{\mu}_i)\). The standardized residual for the covariance between the \(i\)th and \(j\)th observed variables is

\[ \frac{S_{ij} - \hat{\sigma}_{ij}}{\sqrt{\hat{\text{Var}}(S_{ij} - \hat{\sigma}_{ij})}} \]
where

$$\hat{\text{Var}}(S_{ij} - \sigma_{ij}) = \hat{\text{Var}}(S_{ij}) - \hat{\text{Var}}(\sigma_{ij})$$

and $\hat{\text{Var}}(\sigma_{ij})$ is computed using the delta method. Missing values are reported when the computed value of $\hat{\text{Var}}(S_{ij})$ is less than $\hat{\text{Var}}(\sigma_{ij})$. The variances of the raw residuals used in the standardized residual calculations are derived in Hausman (1978).

**Testing standardized parameters**

`estat stdize` provides access to tests on the standardized parameter estimates. `estat stdize` can be used as a prefix to `lincom` (see [R] `lincom`), `nlcom` (see [R] `nlcom`), `test` (see [R] `test`), and `testnl` (see [R] `testnl`).

**Stability of nonrecursive systems**

`estat stable` reports a stability index for nonrecursive systems. The stability index is calculated as the maximum of the modulus of the eigenvalues of $B$. The nonrecursive system is considered stable if the stability index is less than 1.

**Direct, indirect, and total effects**

`estat teffects` reports direct, indirect, and total effects for the fitted model. The direct effects are

$$E_d = [\hat{B} \ \hat{\Gamma}]$$

the total effects are

$$E_t = [(I - \hat{B})^{-1} - I , \ (I - \hat{B})^{-1}\hat{\Gamma}]$$

and the indirect effects are $E_i = E_t - E_d$. The standard errors of the effects are computed using the delta method.

Let $D$ be the diagonal matrix whose elements are the square roots of the diagonal elements of $\hat{\Sigma}$, and let $D_Y$ be the submatrix of $D$ associated with the endogenous variables. Then the standardized effects are

$$\tilde{E}_d = D^{-1}_Y E_d D$$
$$\tilde{E}_i = D^{-1}_Y E_i D$$
$$\tilde{E}_t = D^{-1}_Y E_t D$$

**Predictions**

`predict` computes factor scores and linear predictions.

Factor scores are computed with a linear regression by using the mean vector and variance matrix from the fitted model. For notational convenience, let

$$Z = \begin{pmatrix} z \\ l \end{pmatrix}$$

where

$$z = \begin{pmatrix} y \\ x \end{pmatrix}$$
and
\[ l = \begin{pmatrix} \eta \\ \xi \end{pmatrix} \]

The fitted mean of \( Z \) is
\[ \hat{\mu}_Z = \begin{pmatrix} \hat{\mu}_z \\ \hat{\mu}_l \end{pmatrix} \]

and fitted variance of \( Z \) is
\[ \hat{\Sigma}_Z = \begin{pmatrix} \hat{\Sigma}_{zz} & \hat{\Sigma}_{zl} \\ \hat{\Sigma}_{zl} & \hat{\Sigma}_{ll} \end{pmatrix} \]

The factor scores are computed as
\[ \tilde{l} = \begin{pmatrix} \tilde{\eta} \\ \tilde{\xi} \end{pmatrix} = \hat{\Sigma}'_{zl} \hat{\Sigma}_{zz} \hat{\mu}_z + \hat{\mu}_l \]

The linear prediction for the endogenous variables in the \( t \)th observation is computed as
\[ \tilde{Y}_t = \hat{B} \tilde{Y}_t + \hat{\Gamma} \tilde{X}_t + \hat{\alpha} \]

where
\[ \tilde{Y}_t = \begin{pmatrix} y_t \\ \tilde{\eta} \end{pmatrix} \]

and
\[ \tilde{X}_t = \begin{pmatrix} x_t \\ \tilde{\xi} \end{pmatrix} \]

References


Also see

[SEM] sem — Structural equation model estimation command
Description

`nlcom` is a postestimation command for use after `sem`, `gsem`, and other Stata estimation commands.

`nlcom` computes point estimates, standard errors, z statistics, p-values, and confidence intervals for (possibly) nonlinear combinations of the estimated parameters. See [R] `nlcom`.

Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Nonlinear combinations of parameters

Syntax

```
nlcom exp [ , options ]
```

Options

See `Options` in [R] `nlcom`.

Remarks and examples

See [SEM] Example 42g.

`nlcom` works in the metric of SEM, which is to say path coefficients, variances, and covariances. If you want to frame your nonlinear combinations in terms of standardized coefficients and correlations and you fit the model with `sem`, not `gsem`, then prefix `nlcom` with `estat stdize:`; see [SEM] `estat stdize`.

Technical note

`estat stdize:` is, strictly speaking, unnecessary because everywhere you wanted a standardized coefficient or correlation, you could just type the formula. If you did that, you would get the same results except for numerical precision. The answer produced with the `estat stdize:` prefix will be a little more accurate because `estat stdize:` is able to substitute an analytic derivative in one part of the calculation where `nlcom`, doing the whole thing itself, would be forced to use a numeric derivative.
Stored results

See *Stored results* in [R] `nlcom`.

Also see

[R] `nlcom` — Nonlinear combinations of estimators

[SEM] `estat stdize` — Test standardized parameters

[SEM] `lincom` — Linear combinations of parameters

[SEM] `test` — Wald test of linear hypotheses

[SEM] `testnl` — Wald test of nonlinear hypotheses

[SEM] Example 42g — One- and two-level mediation models (multilevel)
predict after gsem — Generalized linear predictions, etc.

Description

`predict` is a standard postestimation command of Stata. This entry concerns use of `predict` after `gsem`. See [SEM] `predict after sem` if you fit your model with `sem`.

`predict` after `gsem` creates new variables containing observation-by-observation values of estimated observed response variables, linear predictions of observed response variables, latent class probabilities, or endogenous or exogenous continuous latent variables.

Menu

Statistics > SEM (structural equation modeling) > Predictions

Syntax

Syntax for predicting observed endogenous outcomes and other statistics

```
predict [type] { stub* | newvarlist } [if] [in] [, statistic options]
```

Syntax for obtaining estimated continuous latent variables and their standard errors

```
predict [type] { stub* | newvarlist } [if] [in] , lstatistic [loptions]
```

Syntax for obtaining ML scores

```
predict [type] { stub* | newvarlist } [if] [in] , scores
```

The default is to predict observed endogenous variables with empirical Bayes means predictions of the continuous latent variables. If the model includes a categorical latent variable, the default is class-specific predictions of the observed endogenous variables.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mu</code></td>
<td>expected value of <code>depvar</code>; the default</td>
</tr>
<tr>
<td><code>pr</code></td>
<td>probability (synonym for <code>mu</code> when (\mu) is a probability)</td>
</tr>
<tr>
<td><code>eta</code></td>
<td>expected value of linear prediction of <code>depvar</code></td>
</tr>
<tr>
<td><code>density</code></td>
<td>density function at <code>depvar</code></td>
</tr>
<tr>
<td><code>distribution</code></td>
<td>distribution function at <code>depvar</code></td>
</tr>
<tr>
<td><code>survival</code></td>
<td>survivor function at <code>depvar</code></td>
</tr>
<tr>
<td><code>expression(exp)</code></td>
<td>calculate prediction using <code>exp</code></td>
</tr>
<tr>
<td><code>classpr</code></td>
<td>latent class probability</td>
</tr>
<tr>
<td><code>classposteriorpr</code></td>
<td>posterior latent class probability</td>
</tr>
</tbody>
</table>
### predict after gsem — Generalized linear predictions, etc.

#### options

<table>
<thead>
<tr>
<th>Main</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>conditional(ctype)</code></td>
<td>compute statistic conditional on estimated continuous latent variables; default is <code>conditional(ebmeans)</code></td>
</tr>
<tr>
<td><code>marginal</code></td>
<td>compute statistic marginally with respect to the latent variables</td>
</tr>
<tr>
<td><code>pmarginal</code></td>
<td>compute mu marginally with respect to the posterior latent class probabilities</td>
</tr>
<tr>
<td><code>nooffset</code></td>
<td>make calculation ignoring offset or exposure</td>
</tr>
<tr>
<td><code>† outcome(depvar [#])</code></td>
<td>specify observed response variable (default all)</td>
</tr>
<tr>
<td><code>∗ class(lclspec)</code></td>
<td>specify latent class (default all)</td>
</tr>
</tbody>
</table>

#### Integration

<table>
<thead>
<tr>
<th><code>int_options</code></th>
<th>integration options</th>
</tr>
</thead>
</table>

† `outcome(depvar #)` is allowed only if `depvar` has family `multinomial`, `ordinal`, or `bernoulli`. Predicting other generalized responses requires specifying only `outcome(depvar)`. `outcome(depvar #)` may also be specified as `outcome(#.depvar)` or `outcome(depvar #)`.

`outcome(depvar #3)` means the third outcome value. `outcome(depvar #3)` would mean the same as `outcome(depvar #)` if outcomes were 1, 3, and 4.

∗ `class(lclspec)` is allowed only for models with categorical latent variables. For models with one categorical latent variable, `lclspec` can be a class value, such as `class(2)` or its equivalent factor-variable notation `class(2.C)`, assuming the categorical latent variable is C. For models with two or more categorical latent variables, `lclspec` may only be in factor-variable notation, such as `class(2.C#1.D)` for categorical latent variables C and D.

#### ctype

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ebmeans</code></td>
<td>empirical Bayes means of latent variables; the default</td>
</tr>
<tr>
<td><code>ebmodes</code></td>
<td>empirical Bayes modes of latent variables</td>
</tr>
<tr>
<td><code>fixedonly</code></td>
<td>prediction for the fixed portion of the model only</td>
</tr>
</tbody>
</table>

#### lstatistic

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>latent</code></td>
<td>empirical Bayes prediction of all latent variables</td>
</tr>
<tr>
<td><code>latent(varlist)</code></td>
<td>empirical Bayes prediction of specified latent variables</td>
</tr>
</tbody>
</table>

#### loptions

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ebmeans</code></td>
<td>empirical Bayes means of latent variables; the default</td>
</tr>
<tr>
<td><code>ebmodes</code></td>
<td>empirical Bayes modes of latent variables</td>
</tr>
<tr>
<td>`se(stub*</td>
<td>newvarlist)`</td>
</tr>
</tbody>
</table>

#### Integration

<table>
<thead>
<tr>
<th><code>int_options</code></th>
<th>integration options</th>
</tr>
</thead>
</table>
int_options

| intpoints(#) | use # quadrature points to compute marginal predictions and empirical Bayes means |
| iterate(#) | set maximum number of iterations in computing statistics involving empirical Bayes estimators |
| tolerance(#) | set convergence tolerance for computing statistics involving empirical Bayes estimators |

Options

**Main**

- **mu**, the default, calculates the expected value of the outcomes.
- **pr** calculates predicted probabilities and is a synonym for **mu**. This option is available only for multinomial, ordinal, and Bernoulli outcomes.
- **eta** calculates the fitted linear prediction.
- **density** calculates the density function. This prediction is computed using the current values of the observed variables, including the dependent variable.
- **distribution** calculates the distribution function. This prediction is computed using the current values of the observed variables, including the dependent variable. This option is not allowed for multinomial outcomes.
- **survival** calculates the survivor function. This prediction is computed using the current values of the observed variables, including the dependent variable. This option is only allowed for exponential, gamma, loglogistic, lognormal, and Weibull outcomes.
- **expression(exp)** specifies the prediction as an expression. *exp* is any valid Stata expression, but the expression must contain a call to one of the two special functions unique to this option:
  1. **mu(outcome)**: The **mu()** function specifies the calculation of the mean prediction for **outcome**. If **mu()** is specified without **outcome**, the mean prediction for the first outcome is implied.
  2. **pr(outcome)**: The **pr()** function is a synonym for **mu(outcome)** when **outcome** identifies a multinomial, ordinal, or Bernoulli outcome.
- **classpr** calculates predicted probabilities for each latent class.
- **classposteriorpr** calculates predicted posterior probabilities for each latent class. The posterior probabilities are a function of the latent class predictors and the fitted outcome densities.

- **conditional(ctype)**, **marginal**, and **pmarginal** specify how latent variables are handled in computing **statistic**.
  - **conditional()** specifies that **statistic** will be computed conditional on specified or estimated continuous latent variables.
conditional(ebmeans), the default, specifies that empirical Bayes means be used as the estimates of the latent variables. These estimates are also known as posterior mean estimates of the latent variables.

conditional(ebmodes) specifies that empirical Bayes modes be used as the estimates of the latent variables. These estimates are also known as posterior mode estimates of the latent variables.

conditional(fixedonly) specifies that all latent variables be set to zero, equivalent to using only the fixed portion of the model.

marginal specifies that the predicted statistic be computed marginally with respect to the latent variables.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates. They are also required by margins for models with continuous latent variables.

For models with continuous latent variables, the statistic is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

For models with categorical latent variables, \( \mu \) is the only supported statistic. The overall expected value of each outcome is predicted by combining the class-specific expected values using the latent class probabilities.

pmarginal specifies that the overall expected value of each outcome be predicted by combining the class-specific expected values using the posterior latent class probabilities. This option is allowed only with the default statistic, \( \mu \).

nooffset is relevant only if option offset() or exposure() was specified at estimation time. nooffset specifies that offset() or exposure() be ignored, which produces predictions as if all subjects had equal exposure.

outcome(depvar [#]) specifies that predictions for depvar be calculated. Predictions for all observed response variables are computed by default. If depvar is a multinomial or an ordinal outcome, then # optionally specifies which outcome level to predict.

class(lclspec) specifies that predictions for latent class lclspec be calculated. Predictions for all latent classes are computed by default. For models with one categorical latent variable, such as C, lclspec can be a class value, such as \( \text{class}(2) \) or its equivalent factor-variable notation, \( \text{class}(2.C) \). For models with two or more categorical latent variables, such as C and D, lclspec may only be in factor-variable notation, such as \( \text{class}(2.C) \) or \( \text{class}(2.C#1.D) \).

latent and latent(varlist) specify that the continuous latent variables be estimated using empirical Bayes predictions. By default or if the ebmeans option is specified, empirical Bayes means are computed. With the ebmodes option, empirical Bayes modes are computed.

latent requests empirical Bayes estimates for all latent variables.

latex(varlist) requests empirical Bayes estimates for the specified latent variables.

ebmeans specifies that empirical Bayes means be used to predict the latent variables.

ebmodes specifies that empirical Bayes modes be used to predict the latent variables.

se(stub* | newvarlist) calculates standard errors of the empirical Bayes estimators and stores the result in newvarlist. This option requires the latent or latent() option.

scores calculates the scores for each coefficient in e(b). This option requires a new variable list of length equal to the number of columns in e(b). Otherwise, use stub* to have predict generate enumerated variables with prefix stub.
Integration

\texttt{intpoints(\#)} specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

\texttt{iterate(\#)} specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

\texttt{tolerance(\#)} specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Remarks and examples

Out-of-sample prediction is allowed for all \texttt{predict} options except \texttt{scores}.

\texttt{predict} has two ways of specifying the names of the variables to be created:

\begin{verbatim}
. predict stub*, ...
\end{verbatim}

or

\begin{verbatim}
. predict firstname secondname ..., ...
\end{verbatim}

The first creates variables named \texttt{stub1}, \texttt{stub2}, \ldots. The second creates variables with names that you specify. We strongly recommend using the \texttt{stub*} syntax when creating multiple variables because you have no way of knowing the order in which to specify the individual variable names to correspond to the order in which \texttt{predict} will make the calculations. If you use \texttt{stub*}, the variables will be labeled and you can rename them.

The second syntax is useful when you create one variable and specify \texttt{outcome()}, \texttt{expression()}, \texttt{class()}, or \texttt{latent()}.

See [SEM] Intro 7, [SEM] Example 28g, [SEM] Example 29g, [SEM] Example 50g, and [SEM] Example 52g.

Reference


Also see

[SEM] \texttt{gsem} — Generalized structural equation model estimation command

[SEM] \texttt{gsem postestimation} — Postestimation tools for gsem

[SEM] \texttt{Intro 7} — Postestimation tests and predictions

[SEM] \texttt{Example 28g} — One-parameter logistic IRT (Rasch) model

[SEM] \texttt{Example 29g} — Two-parameter logistic IRT model

[SEM] \texttt{Example 50g} — Latent class model

[SEM] \texttt{Example 52g} — Latent profile model

[SEM] \texttt{Methods and formulas for gsem} — Methods and formulas for gsem
**Title**

**predict after sem —** Factor scores, linear predictions, etc.

### Description

**predict** is a standard postestimation command of Stata. This entry concerns use of **predict after sem**. See [SEM] **predict after gsem** if you fit your model with **gsem**.

**predict after sem** creates new variables containing observation-by-observation values of estimated factor scores (meaning predicted values of latent variables) and predicted values for latent and observed endogenous variables. Out-of-sample prediction is allowed.

When **predict** is used on a model fit by **sem** with the **group()** option, results are produced with the appropriate group-specific estimates. Out-of-sample prediction is allowed; missing values are filled in for groups not included at the time the model was fit.

**predict** allows two syntaxes. You can type

```
.predict stub*, ...
```

to create variables named **stub1**, **stub2**, ..., or you can type

```
.predict var1 var2..., ...
```

to create variables named **var1**, **var2**, ....

**predict** may not be used with summary statistics data.

### Menu

Statistics > SEM (structural equation modeling) > Predictions

### Syntax

```
predict [type] { stub* newvarlist } [if] [in] [, options ]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>xb</strong></td>
<td>linear prediction for all OEn variables; the default</td>
</tr>
<tr>
<td><strong>xb(varlist)</strong></td>
<td>linear prediction for specified OEn variables</td>
</tr>
<tr>
<td><strong>xblatent</strong></td>
<td>linear prediction for all LEn variables</td>
</tr>
<tr>
<td><strong>xblatent(varlist)</strong></td>
<td>linear prediction for specified LEn variables</td>
</tr>
<tr>
<td><strong>latent</strong></td>
<td>factor scores for all latent variables</td>
</tr>
<tr>
<td><strong>latent(varlist)</strong></td>
<td>factor scores for specified latent variables</td>
</tr>
<tr>
<td><strong>scores</strong></td>
<td>calculate first derivative of the log likelihood</td>
</tr>
</tbody>
</table>

Key: OEn = observed endogenous; LEn = latent endogenous
Options

xb calculates the linear prediction for all observed endogenous variables in the model. xb is the default if no option is specified.

xb(varlist) calculates the linear prediction for the variables specified, all of which must be observed endogenous variables.

xblatent and xblatent(varlist) calculate the linear prediction for all or the specified latent endogenous variables, respectively.

latent and latent(varlist) calculate the factor scores for all or the specified latent variables, respectively. The calculation method is an analog of regression scoring; namely, it produces the means of the latent variables conditional on the observed variables used in the model. If missing values are found among the observed variables, conditioning is on the variables with observed values only.

scores is for use by programmers. It provides the first derivative of the observation-level log likelihood with respect to the parameters.

Programmers: In single-group sem, each parameter that is not constrained to be 0 has an associated equation. As a consequence, the number of equations, and hence the number of score variables created by predict, may be large.

Remarks and examples

See [SEM] Example 14.

Factor scoring for latent variables can be interpreted as a form of missing-value imputation—think of each latent variable as an observed variable that has only missing values.

When latent variables are present in the model, linear predictions from predict, xb are computed by substituting the factor scores in place of each latent variable before computing the linear combination of coefficients. This method will lead to inconsistent coefficient estimates when the factor score contains measurement error; see Bollen (1989, 305–306).

Reference


Also see

[SEM] Example 14 — Predicted values
[SEM] Methods and formulas for sem — Methods and formulas for sem
[SEM] sem postestimation — Postestimation tools for sem
Description

`sem` fits structural equation models. Even when you use the SEM Builder, you are using the `sem` command.

Menu

Statistics > SEM (structural equation modeling) > Model building and estimation

Syntax

```
sem paths [if] [in] [weight] [ , options ]
```

where `paths` are the paths of the model in command-language path notation; see [SEM] sem and gsem path notation.

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>model_description_options</code></td>
<td>fully define, along with <code>paths</code>, the model to be fit</td>
</tr>
<tr>
<td><code>group_options</code></td>
<td>fit model for different groups</td>
</tr>
<tr>
<td><code>ssd_options</code></td>
<td>for use with summary statistics data</td>
</tr>
<tr>
<td><code>estimation_options</code></td>
<td>method used to obtain estimation results</td>
</tr>
<tr>
<td><code>reporting_options</code></td>
<td>reporting of estimation results</td>
</tr>
<tr>
<td><code>syntax_options</code></td>
<td>controlling interpretation of syntax</td>
</tr>
</tbody>
</table>

Time-series operators are allowed.

`bootstrap`, `by`, `jackknife`, `permute`, `statsby`, and `svy` are allowed; see [U] 11.1.10 Prefix commands.

Weights are not allowed with the bootstrap prefix; see [R] `bootstrap`.

`vce()` and weights are not allowed with the `svy` prefix; see [SVY] `svy`.

`fweight`, `iweight`, and `pweight` are allowed; see [U] 11.1.6 weight.

Also see [SEM] `sem postestimation` for features available after estimation.

Options

`model_description_options` describe the model to be fit. The model to be fit is fully specified by `paths`—which appear immediately after `sem`—and the options `covariance()`, `variance()`, and `means()`. See [SEM] sem model description options and [SEM] sem and gsem path notation.
group_options allow the specified model to be fit for different subgroups of the data, with some parameters free to vary across groups and other parameters constrained to be equal across groups. See [SEM] sem group options.

ssd_options allow models to be fit using summary statistics data (SSD), meaning data on means, variances (standard deviations), and covariances (correlations). See [SEM] sem ssd options.

estimation_options control how the estimation results are obtained. These options control how the standard errors (VCE) are obtained and control technical issues such as choice of estimation method. See [SEM] sem estimation options.

reporting_options control how the results of estimation are displayed. See [SEM] sem reporting options.

syntax_options control how the syntax that you type is interpreted. See [SEM] sem and gsem syntax options.

Remarks and examples

For a readable explanation of what \texttt{sem} can do and how to use it, see any of the intro sections. You might start with [SEM] Intro 1.

For examples of \texttt{sem} in action, see any of the example sections. You might start with [SEM] Example 1.

For detailed syntax and descriptions, see the references below.

Remarks on three advanced topics are presented under the following headings:

- Default normalization constraints
- Default covariance assumptions
- How to solve convergence problems

Default normalization constraints

\texttt{sem} applies the following rules as necessary to identify the model:

1. \texttt{means(1: LatentExogenous@0)}
   \texttt{sem} constrains all latent exogenous variables to have mean 0. When the \texttt{group()} option is specified, this rule is applied to the first group only.

2. \texttt{(LatentEndogenous <- _cons@0)}
   \texttt{sem} sets all latent endogenous variables to have intercept 0.

3. \texttt{(first <- Latent@1)}
   If latent variable \texttt{Latent} is measured by observed endogenous variables, then \texttt{sem} sets the path coefficient of \texttt{(first<-Latent)} to be 1; \texttt{first} is the first observed endogenous variable.

4. \texttt{(First<-Latent@1)}
   If item 3 does not apply—if latent variable \texttt{Latent} is measured by other latent endogenous variables only—then \texttt{sem} sets the path coefficient of \texttt{First<-Latent} to be 1; \texttt{First} is the first latent variable.

The above constraints are applied as needed. Here are the available overrides:

1. To override the normalization constraints, specify your own constraints. Most normalization constraints are added by \texttt{sem} as needed. See \texttt{How sem (gsem) solves the problem for you} under Identification 2: Normalization constraints (anchoring) in [SEM] Intro 4.
2. To override means() constraints, you must use the means() option to free the parameter. To override that the mean of latent exogenous variable MyLatent has mean 0, specify the means(MyLatent) option. See [SEM] sem and gsem path notation.

3. To override constrained path coefficients from _cons, such as (LatentEndogenous <- _cons@0), you must explicitly specify the path without a constraint (LatentEndogenous <- _cons). See [SEM] sem and gsem path notation.

Default covariance assumptions

`sem` assumes the following covariance structure:

1. covstructure(_Ex, unstructured)
   All exogenous variables (observed and latent) are assumed to be correlated with each other.

2. covstructure(e._En, diagonal)
   The error variables associated with all endogenous variables are assumed to be uncorrelated with each other.

You can override these assumptions by

1. Constraining or specifying the relevant covariance to allow e.y and e.Ly to be correlated (specify the covariance(e.y*e.Ly) option); see [SEM] sem model description options.
2. Using the covstructure() option; see [SEM] sem and gsem option covstructure().

How to solve convergence problems

See [SEM] Intro 12.

Stored results

`sem` stores the following in `e()`: Scalars

- `e(N)` number of observations
- `e(N_clust)` number of clusters
- `e(N_groups)` number of groups
- `e(N_missing)` number of missing values in the sample for method(mlmv)
- `e(ll)` log likelihood of model
- `e(df_m)` model degrees of freedom
- `e(df_b)` baseline model degrees of freedom
- `e(df_s)` saturated model degrees of freedom
- `e(chi2_ms)` test of target model against saturated model
- `e(df_ms)` degrees of freedom for `e(chi2_ms)`
- `e(p_ms)` p-value for `e(chi2_ms)`
- `e(chi2sb_ms)` Satorra–Bentler scaled test of target model against saturated model
- `e(psb_ms)` p-value for `e(chi2sb_ms)`
- `e(sbc_ms)` Satorra–Bentler correction factor for `e(chi2sb_ms)`
- `e(chi2_bs)` test of baseline model against saturated model
- `e(df_bs)` degrees of freedom for `e(chi2_bs)`
- `e(p_bs)` p-value for `e(chi2_bs)`
- `e(chi2sb_bs)` Satorra–Bentler scaled test of baseline model against saturated model
- `e(psb_bs)` p-value for `e(chi2sb_bs)`
- `e(sbc_bs)` Satorra–Bentler correction factor for `e(chi2_bs)`
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
**sem — Structural equation model estimation command**

- `e(converged)` 1 if target model converged, 0 otherwise
- `e(critvalue)` log likelihood or discrepancy of fitted model
- `e(critvalue_b)` log likelihood or discrepancy of baseline model
- `e(critvalue_s)` log likelihood or discrepancy of saturated model
- `e(modelmeans)` 1 if fitting means and intercepts, 0 otherwise

**Macros**
- `e(cmd)` sem
- `e(cmdline)` command as typed
- `e(data)` raw or ssd if SSD were used
- `e(vtype)` weight type
- `e(wexp)` weight expression
- `e(title)` title in estimation output
- `e(clustvar)` name of cluster variable
- `e(vce)` vcetype specified in vce()
- `e(vcetype)` title used to label Std. Err.
- `e(method)` estimation method: ml, mlmv, or adf
- `e(technique)` maximization technique
- `e(properties)` b V
- `e(estat_cmd)` program used to implement estat
- `e(predict)` program used to implement predict
- `e(lyvars)` names of latent y variables
- `e(oyvars)` names of observed y variables
- `e(lxvars)` names of latent x variables
- `e(oxvars)` names of observed x variables
- `e(groupvar)` name of group variable
- `e(xconditional)` empty if nocluestimate specified, xconditional otherwise
- `e(marginsnotok)` predictions not allowed by margins
- `e(marginsdefault)` default predict() specification for margins

**Matrices**
- `e(b)` parameter vector
- `e(b_std)` standardized parameter vector
- `e(b_pclass)` parameter class
- `e(Cns)` constraints matrix
- `e(admissible)` admissibility of Σ, Ψ, Φ
- `e(ilog)` iteration log (up to 20 iterations)
- `e(gradient)` gradient vector
- `e(V)` covariance matrix of the estimators
- `e(V_std)` standardized covariance matrix of the estimators
- `e(V_modelbased)` model-based variance
- `e(nobs)` vector with number of observations per group
- `e(groupvalue)` vector of group values of e(groupvar)

**Functions**
- `e(sample)` marks estimation sample (not with SSD)

---

**References**


Also see

[SEM] Intro 1 — Introduction

[SEM] sem and gsem path notation — Command syntax for path diagrams

[SEM] sem path notation extensions — Command syntax for path diagrams

[SEM] sem model description options — Model description options

[SEM] sem group options — Fitting models on different groups

[SEM] sem ssd options — Options for use with summary statistics data

[SEM] sem estimation options — Options affecting estimation

[SEM] sem reporting options — Options affecting reporting of results

[SEM] sem and gsem syntax options — Options affecting interpretation of syntax

[SEM] sem postestimation — Postestimation tools for sem

[SEM] Methods and formulas for sem — Methods and formulas for sem

[SVY] svy estimation — Estimation commands for survey data
sem and gsem option constraints() — Specifying constraints

Description

Constraints refer to constraints to be imposed on the estimated parameters of a model. These constraints usually come in one of three forms:

1. Constraints that a parameter such as a path coefficient or variance is equal to a fixed value such as 1.
2. Constraints that two or more parameters are equal.
3. Constraints that two or more parameters are related by a linear equation.

It is usually easier to specify constraints with sem’s and gsem’s path notation; see [SEM] sem and gsem path notation.

sem’s and gsem’s constraints() option provides an alternative way of specifying constraints.

Syntax

```
sem ... [ , ... constraints(# [# ...]) ... ]
gsem ... [ , ... constraints(# [# ...]) ... ]
```

where # are constraint numbers. Constraints are defined by the constraint command; see [R] constraint.

Remarks and examples

Remarks are presented under the following headings:

Use with sem

Use with gsem

Also see [R] constraint.

Use with sem

There is only one case where constraints() might be easier to use with sem instead of specifying constraints in the path notation. You wish to specify that two or more parameters are related and then decide you would like to fix the value at which they are related.

For example, if you wanted to specify that parameters are equal, you could type

```
. sem ... (y1 <- x@c1) (y2 <- x@c1) (y3 <- x@c1) ...
```

Using the path notation, you can specify more general relationships, too, such as

```
. sem ... (y1 <- x@c1) (y2 <- x@(2*c1)) (y3 <- x@(3*c1+1)) ...
```
Say you now decide you want to fix \( c_1 \) at value 1. Using the path notation, you modify what you previously typed:

```
  . sem ... (y1 <- x@1) (y2 <- x@2) (y3 <- x@4) ...
```

Alternatively, you could do the following:

```
  . constraint 1 _b[y2:x] = 2*_b[y1:x]
  . constraint 2 _b[y3:x] = 3*_b[y1:x] + 1
  . sem ..., ... constraints(1 2)
  . constraint 3 _b[y1:x] = 1
  . sem ..., ... constraints(1 2 3)
```

### Use with gsem

Gamma regression can produce exponential regression estimates if you constrain the log of the scale parameter to 0. Parameters associated with particular generalized linear families, such as scalar parameters, cutpoints, and the like, cannot be constrained using the @ notation in paths. You must use Stata’s constraints.

Say we wish to fit the model \( y \leftarrow x_1 \) with exponential regression. We admit that we do not remember the name under which \texttt{gsem} stores the scalar parameter, so first we type

```
  . gsem (y <- x1, gamma), noestimate
```

From the output, we quickly discover that the log of the scale parameter is stored as \_b[/y:logs]. With that information, to obtain the constrained results, we type

```
  . constraint 1 _b[/y:logs] = 0
  . gsem (y <- x1, gamma), constraints(1)
```

### Also see

- \texttt{SEM sem} — Structural equation model estimation command
- \texttt{SEM gsem} — Generalized structural equation model estimation command
- \texttt{SEM sem and gsem path notation} — Command syntax for path diagrams
- \texttt{SEM gsem model description options} — Model description options
- \texttt{SEM sem model description options} — Model description options
- \texttt{R constraint} — Define and list constraints
**sem and gsem option covstructure() — Specifying covariance restrictions**

### Description

Option `covstructure()` provides a sometimes convenient way to constrain the covariances of your model.

Alternatively or in combination, you can place constraints on the covariances by using the standard path notation, such as

```
  . sem ..., ... cov(name1*name2@c1 name3*name4@c1) ...
  . gsem ..., ... cov(name1*name2@c1 name3*name4@c1) ...
```

See [SEM] `sem and gsem path notation`.

If you are using `sem`, also see [SEM] `sem path notation extensions` for documentation on how the syntax of `covstructure()` is modified when the `group()` option is specified.

If you are using `gsem`, also see [SEM] `gsem path notation extensions` for documentation on how the syntax of `covstructure()` is modified when the `group()` option or the `lclass()` options are specified.

### Syntax

```
sem ... [, ... covstructure(variables, structure) ... ]
gsem ... [, ... covstructure(variables, structure) ... ]
```

where `variables` is one of

1. a list of (a subset of the) exogenous variables (`sem`) or latent exogenous variables (`gsem`) in your model, for instance,

   ```
   . sem ..., ... covstruct(x1 x2, structure)
   . sem ..., ... covstruct(L1 L2, structure)
   . gsem ..., ... covstruct(L1 L2, structure)
   ```
2. `_0Ex`, meaning all observed exogenous variables in your model (`sem` only)
3. `_LEx`, meaning all latent exogenous variables in your model (including any multilevel latent exogenous variables in the case of `gsem`)
4. `_Ex`, meaning all exogenous variables in your model (`sem` only)

or where `variables` is one of

1. a list of (a subset of the) error variables in your model, for example,

   ```
   . sem ..., ... covstruct(e.y1 e.y2 e.Aspect, structure)
   ```
2. `e._0En`, meaning all error variables associated with observed endogenous variables in your model
3. `e._LEn`, meaning all error variables associated with latent endogenous variables in your model
4. `e._En`, meaning all error variables in your model
and where \textit{structure} is

<table>
<thead>
<tr>
<th>structure</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{diagonal}</td>
<td>all variances unrestricted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>all covariances fixed at 0</td>
<td></td>
</tr>
<tr>
<td>\textit{unstructured}</td>
<td>all variances unrestricted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>all covariances unrestricted</td>
<td></td>
</tr>
<tr>
<td>\textit{identity}</td>
<td>all variances equal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>all covariances fixed at 0</td>
<td></td>
</tr>
<tr>
<td>\textit{exchangeable}</td>
<td>all variances equal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>all covariances equal</td>
<td></td>
</tr>
<tr>
<td>\textit{zero}</td>
<td>all variances fixed at 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>all covariances fixed at 0</td>
<td></td>
</tr>
<tr>
<td>\textit{pattern(matname)}</td>
<td>covariances (variances) unrestricted if \text{matname}[i, j] \geq .</td>
<td>(1)</td>
</tr>
<tr>
<td></td>
<td>covariances (variances) equal if \text{matname}[i, j] = \text{matname}[k, l]</td>
<td></td>
</tr>
<tr>
<td>\textit{fixed(matname)}</td>
<td>covariances (variances) unrestricted if \text{matname}[i, j] \geq .</td>
<td>(2)</td>
</tr>
<tr>
<td></td>
<td>covariances (variances) fixed at \text{matname}[i, j] otherwise</td>
<td></td>
</tr>
</tbody>
</table>

Notes:

1. Only elements in the lower triangle of \textit{matname} are used. All values in \textit{matname} are interpreted as the floor() of the value if noninteger values appear. Row and column stripes of \textit{matname} are ignored.

2. Only elements on the lower triangle of \textit{matname} are used. Row and column stripes of \textit{matname} are ignored.

Option

covstructure(\textit{variables}, \textit{structure}) is used either to modify the covariance structure among the exogenous variables of your model or to modify the covariance structure among the error variables of your model.

You may specify the covstructure() option multiple times.

The default covariance structure for exogenous variables is covstructure(_Ex, unstructured) for \textit{sem}. There is no simple way in this notation to write the default for \textit{gsem}.

The default covariance structure for error variables is covstructure(e._En, diagonal) for \textit{sem} and \textit{gsem}. 
Remarks and examples

See [SEM] Example 17.

Standard linear structural equation modeling allows covariances among exogenous variables, both latent and observed, and allows covariances among the error variables. Covariances between exogenous variables and error variables are disallowed (assumed to be 0).

Some authors refer to the covariances among the exogenous variables in standard SEMs as matrix $\Phi$ and to the covariances among the error variables as matrix $\Psi$.

Also see

[SEM] sem — Structural equation model estimation command
[SEM] gsem — Generalized structural equation model estimation command
[SEM] sem and gsem path notation — Command syntax for path diagrams
[SEM] Example 17 — Correlated uniqueness model
sem and gsem option from() — Specifying starting values

Description

See [SEM] Intro 12 for a description of starting values.

Starting values are usually not specified. When there are convergence problems, it is often necessary to specify starting values. You can specify starting values by using

1. suboption init() as described in [SEM] sem and gsem path notation, or by using
2. option from() as described here.

Option from() is especially convenient for using the solution of one model as starting values for another.

Syntax

\{sem|gsem\} ... [ , ... from(matname[, skip]) ] ...
\{sem|gsem\} ... [ , ... from(svlist) ] ...

where matname is the name of a Stata matrix and
where svlist is a space-separated list of the form
	\eqname:name = #

Option

skip is an option of from(matname). It specifies to ignore any parameters in matname that do not appear in the model being fit. If this option is not specified, the existence of such parameters causes sem (gsem) to issue an error message.

Remarks and examples

Remarks are presented under the following headings:

Syntax 1, especially useful when dealing with convergence problems
Syntax 2, seldom used

Option from() can be used with sem or gsem. We illustrate below using sem.
Syntax 1, especially useful when dealing with convergence problems

Say you are attempting to fit

```
.semm your_full_model, ...
```

and are having difficulty with convergence. Following the advice in [SEM] Intro 12, you have simplified your model,

```
.semm your_simple_model, ...
```

and that does converge. You now want to use those values as starting values for the full model. Let’s imagine that there are 47 estimated parameters in the simple model.

Using the standard `init()` method for specifying starting values, you now have a real job in front of you. You have to type your full model, find all the places where you want to add starting values, and add an `init()` suboption. Just a piece of your full model might read

```
... (y<-L1 L2) (L1->x1 x2) (L2->x3 L4) ... 
```

and you need to modify that to read

```
... (y<-(L1, init(14.283984)) L2) ///
   (L1->(x1, init(2.666532)) (x2, init(-6.39499))) ///
   (L2->x3 L4) ...
```

That change handles 3 of the 47 parameters you need to specify.

There is an easier way. Type

```
.semm your_simple_model, ...
.matrix b = e(b)
.semm your_full_model, ... from(b)
```

Here is how this works:

1. You fit the simple model. sem stores the resulting parameters in `e(b).
2. You store the fitted parameters in Stata matrix `b.
3. To fit your full model, type the model just as you would usually, and add option `from(b). That option tells sem to get any starting values it can from Stata matrix `b. sem does that and then follows its usual logic for producing starting values for the remaining parameters.

Just because you use `from(b) does not mean you cannot specify starting values with `init(). You can even specify starting values for some of the same parameters. Starting values specified by `init() take precedence over those obtained from `from().

Syntax 2, seldom used

In syntax 2, you specify

```
.semm ..., ... from(eqname:name = # eqname:name = # ...) 
```

For instance, you could type

```
.semm ..., ... from(var(X):_cons=10)
```
or you could type

```
.semm ..., ... var(X, init(10))
```

It is usually easier to type the second. See [SEM] sem and gsem path notation.
You may combine the two notations. If starting values are specified for a parameter both ways, those specified by `init()` take precedence.

Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM] `gsem` — Generalized structural equation model estimation command
- [SEM] `sem and gsem path notation` — Command syntax for path diagrams
- [SEM] `gsem model description options` — Model description options
- [SEM] `sem model description options` — Model description options
- [SEM] `gsem estimation options` — Options affecting estimation
- [R] `Maximize` — Details of iterative maximization
Option `reliability()` allows you to specify the fraction of variance not due to measurement error for measurement variables.

Syntax

```
{sem | gsem} ... [ , ... reliability(varname # [ varname # [ ... ]]) ]
```

where `varname` is the name of an observed endogenous variable and `#` is the fraction or percentage of variance not due to measurement error:

- `{sem | gsem} ... , ... reliability(x1 .8 x2 .9)
- `{sem | gsem} ... , ... reliability(x1 80% x2 90%)

Option

`reliability(varname # [ ... ])` specifies the reliability for variable `varname`. Reliability is bounded by 0 and 1 and is equal to

$$1 - \frac{\text{noise variance}}{\text{total variance}}$$

The reliability is assumed to be 1 when not specified.

Remarks and examples

See [SEM] Example 24.

Remarks are presented under the following headings:

- Background
- Dealing with measurement error of exogenous variables
- Dealing with measurement error of endogenous variables
- What can go wrong

Background

Option `reliability()` may be used with `sem` and may be used with `gsem` for Gaussian response variables with the identity link but only in the absence of censoring. We illustrate using `sem`, but we could just as well have used `gsem`.

Variables measured with error have attenuated path coefficients. If we had the model

```
. sem (y<-x)
```
and $x$ were measured with error, then the estimated path coefficient would be biased toward 0. The usual solution to such measurement problems is to find multiple measurements and develop a latent variable from them:

```
.semm (x1 x2 x3<-X) (y<-X)
```

Another solution is available if we know the reliability of $x$. In that case, we can fit the model

```
.semm (x<-X) (y<-X), reliability(x .9)
```

The two solutions can even be combined:

```
.semm (x1 x2 x3<-X) (y<-X), reliability(x1 .9 x2 .8 x3 .9)
```

Even if you do not know the reliability, you can experiment using different but reasonable values for the reliability and thus determine the sensitivity of your estimation results to the measurement problem.

### Dealing with measurement error of exogenous variables

Measurement error is most important when it occurs in exogenous variables, yet the `reliability()` option deals with measurement error of endogenous variables only. By creation of a latent variable, `reliability()` can deal with the measurement error of exogenous variables.

To fit the model $(y<-x)$ where $x$ is measured with error, you must introduce a latent variable corresponding to $x$ measured without error. That is, the model $(y<-x)$ can be converted into the model $(x<-X)$ and $(y<-X)$:

\[
x = \alpha_0 + \beta_0 X + e.x
\]

\[
y = \alpha_1 + \beta_1 X + e.y
\]

To fit this model, you type

```
.semm (x<-X) (y<-X), reliability(x .9)
```

`semm` will introduce a normalization constraint, namely, that the path coefficient $\beta_0$ for $x<-X$ is 1, but that is of no importance. What is important is that the estimate of that path coefficient $\beta_1$ of $y<-X$ is the coefficient that would be obtained from $y<-x$ were $x$ measured without error.

In the above, we specified the measurement part of the model first. Be sure to do that. You might think you could equally well reverse the two terms so that, rather than writing

```
(x<-X) (y<-X)  (correct)
```

you could write

```
(y<-X) (x<-X)  (incorrect)
```

But you cannot do that unless you write

```
(y<-X) (x<-X@1)  (correct)
```

because otherwise results are as if you typed

```
(y<-X@1) (x<-X)  (incorrect)
```
All of that is because \texttt{sem} places its normalization constraint from the latent variable to the first observed endogenous variable. There is no real error if the terms are interchanged except that you will be surprised by the coefficient of 1 for $y<-x$ and (the reciprocal of) the coefficient of interest will be on $x<-x$.

See \textit{How sem (gsem) solves the problem for you} in \texttt{SEM Intro 4} and see \textit{Default normalization constraints} in \texttt{SEM sem}.

### Dealing with measurement error of endogenous variables

When a variable would already be endogenous before you add the \texttt{reliability()} option, it really makes little difference whether you add the \texttt{reliability()} option. That is because endogenous variables are assumed to contain error, and if some of that error is measurement error, it is still just an error. Coefficients will be unchanged by the inclusion of the \texttt{reliability()} option.

Some variances and covariances will change, but the changes are offsetting in the calculation of path coefficients.

What will change are the standardized coefficients should you ask to see them. That is because the variances are changed.

### What can go wrong

Consider a model of $y$ on $x$. Say we fit the model with linear regression. If the $R^2$ of the fit is 0.6, then we know the reliability must be greater than 0.6. $R^2$ measures the fraction of variance of $y$ that is explained by $x$, and the reliability of $x$ measures the fraction of the variance of $x$ that is not due to measurement error. Measurement error is assumed to be pure noise. It is just not possible that we could explain 0.6 of the variance of $y$ by using a variable with reliability of, say, 0.5.

Well, in fact, it is because there is always a chance that the pure noise will correlate with $y$, too. Asymptotically, that probability vanishes, but in finite—especially small—samples, it could happen. Even so, the calculation of the corrected \texttt{SEM} estimates blows up.

If you have convergence problems, you need to check for this. Specify a larger value for the reliability. The problem is you cannot specify a value of 1, and large values such as 0.99999 can lead to a lack of identification. In most cases, you will be able to find a value in between, but the only way to be sure is to remove the \texttt{reliability()} option and, if necessary, simplify your model by removing any intermediary latent variables you had to add because of reliability.

If your model converges without reliability, then your measure of reliability is too low. At this point, we have little useful advice for you. Check whether you have the right value, of course. If you do, then there are two possibilities: either the experts who provided that estimate are wrong or you got unlucky in that the measurement error did just happen to correlate with the rest of your data. You will need to evaluate the chances of that for yourself. In any case, you can experiment with higher values of the reliability and at least provide an idea of the sensitivity of your estimates to differing assumptions.

### Also see

- \texttt{SEM sem} — Structural equation model estimation command
- \texttt{SEM gsem} — Generalized structural equation model estimation command
- \texttt{SEM gsem model description options} — Model description options
- \texttt{SEM sem model description options} — Model description options
- \texttt{SEM Example 24} — Reliability
Description

The command syntax for describing your SEM is fully specified by paths, covariance(), variance(), and means(). How this works is described below.

If you are using \texttt{sem}, also see \cite{SEM} \texttt{sem path notation extensions} for documentation of the \texttt{group()} option for comparing different groups in the data. The syntax of the elements described below is modified when \texttt{group()} is specified.

If you are using \texttt{gsem}, also see \cite{SEM} \texttt{gsem path notation extensions} for documentation on specification of family-and-link for generalized (nonlinear) response variables, specification of multilevel latent variables, specification of categorical latent variables, and specification of multiple-group models. The syntax of the elements described below is modified when the \texttt{group()} option for comparing different groups or the \texttt{lclass()} option for categorical latent variables is specified.

Either way, read this section first.

Syntax

\begin{verbatim}
sem paths ... [, covariance() variance() means()]
gsem paths ... [, covariance() variance() means()]
\end{verbatim}

paths specifies the direct paths between the variables of your model.
The model to be fit is fully described by paths, covariance(), variance(), and means().

Options

covariance() is used to

1. specify that a particular covariance path of your model that usually is assumed to be 0 be estimated,
2. specify that a particular covariance path that usually is assumed to be nonzero is not to be estimated (to be constrained to be 0),
3. constrain a covariance path to a fixed value, such as 0, 0.5, 1, etc., and
4. constrain two or more covariance paths to be equal.

variance() does the same as covariance() except it does it with variances.

means() does the same as covariance() except it does it with means.

Remarks and examples

Path notation is used by the \texttt{sem} and \texttt{gsem} commands to specify the model to be fit, for example,

\begin{verbatim}
  . sem (x1 x2 x3 x4 <- X)
  . gsem (L1 -> x1 x2 x3 x4 x5, logit) (L2 -> x6 x7 x8 x9 x10)
\end{verbatim}
In the path notation,

1. Latent variables are indicated by a name in which at least the first letter is capitalized.
2. Observed variables are indicated by a name in which at least the first letter is lowercased. Observed variables correspond to variable names in the dataset.
3. Error variables, while mathematically a special case of latent variables, are considered in a class by themselves. For \texttt{sem}, every endogenous variable (whether observed or latent) automatically has an error variable associated with it. For \texttt{gsem}, the same is true of Gaussian endogenous variables (and latent variables, which are Gaussian). The error variable associated with endogenous variable name is $e \cdot \text{name}$.
4. Paths between variables are written as
   \[(\text{name1} \leftarrow \text{name2})\]
   or
   \[(\text{name2} \rightarrow \text{name1})\]
   There is no significance to which coding is used.
5. Paths between the same variables can be combined: The paths
   \[(\text{name1} \leftarrow \text{name2}) (\text{name1} \leftarrow \text{name3})\]
   can be combined as
   \[(\text{name1} \leftarrow \text{name2 name3})\]
   or as
   \[(\text{name2 name3} \rightarrow \text{name1})\]
   The paths
   \[(\text{name1} \leftarrow \text{name3}) (\text{name2} \leftarrow \text{name3})\]
   can be combined as
   \[(\text{name1 name2} \leftarrow \text{name3})\]
   or as
   \[(\text{name3} \rightarrow \text{name1 name2})\]
   The paths
   \[(\text{name1} \leftarrow \text{name2 name3})
   (\text{name4} \leftarrow \text{name2 name3})\]
   may be written as
   \[(\text{name1 name4} \leftarrow \text{name2 name3})\]
   or as
   \[(\text{name2 name3} \rightarrow \text{name1 name4})\]
6. Variances and covariances (curved paths) between variables are indicated by options. Variances are indicated by

\[ \ldots, \ldots \text{var}(\text{name1}) \]

Covariances are indicated by

\[ \ldots, \ldots \text{cov}(\text{name1*name2}) \]
\[ \ldots, \ldots \text{cov}(\text{name2*name1}) \]

There is no significance to the order of the names.

The actual names of the options are variance() and covariance(), but they are invariably abbreviated as var() and cov(), respectively.

The var() and cov() options are the same option, so a variance can be typed as

\[ \ldots, \ldots \text{cov}(\text{name1}) \]

and a covariance can be typed as

\[ \ldots, \ldots \text{var}(\text{name1*name2}) \]

7. Variances may be combined, covariances may be combined, and variances and covariances may be combined.

If you have

\[ \ldots, \ldots \text{var}(\text{name1}) \text{var}(\text{name2}) \]

you may code this as

\[ \ldots, \ldots \text{var}(\text{name1 name2}) \]

If you have

\[ \ldots, \ldots \text{cov}(\text{name1*name2}) \text{cov}(\text{name2*name3}) \]

you may code this as

\[ \ldots, \ldots \text{cov}(\text{name1*name2 name2*name3}) \]

All the above combined can be coded as

\[ \ldots, \ldots \text{cov}(\text{name1 name2 name1*name2 name2*name3}) \]

or as

\[ \ldots, \ldots \text{var}(\text{name1 name2 name1*name2 name2*name3}) \]

8. All variables except endogenous variables are assumed to have a variance; it is only necessary to code the var() option if you wish to place a constraint on the variance or specify an initial value. See items 11, 12, 13, and 16 below. (In gsem, the variance and covariances of observed exogenous variables are not estimated and thus var() cannot be used with them.)

Endogenous variables have a variance, of course, but that is the variance implied by the model. If \text{name} is an endogenous variable, then var(\text{name}) is invalid. The error variance of the endogenous variable is var(e.\text{name}).

9. Variables mostly default to being correlated:

a. All exogenous variables are assumed to be correlated with each other, whether observed or latent.

b. Endogenous variables are never directly correlated, although their associated error variables can be.

c. All error variables are assumed to be uncorrelated with each other.
You can override these defaults on a variable-by-variable basis with the `cov()` option.

To assert that two variables are uncorrelated that otherwise would be assumed to be correlated, constrain the covariance to be 0:

\[
\ldots, \ldots \text{cov}(\text{name1*name2@0})
\]

To allow two variables to be correlated that otherwise would be assumed to be uncorrelated, simply specify the existence of the covariance:

\[
\ldots, \ldots \text{cov}(\text{name1*name2})
\]

This latter is especially commonly done with errors:

\[
\ldots, \ldots \text{cov}(\text{e.name1*e.name2})
\]

(In gsem, you may not use the `cov()` option with observed exogenous variables. You also may not use `cov()` with error terms associated with family Gaussian, link log.)

10. Means of variables are indicated by the following option:

\[
\ldots, \ldots \text{means(name)}
\]

Variables mostly default to having nonzero means:

a. All observed exogenous variables are assumed to have nonzero means. In sem, the means can be constrained using the `means()` option, but only if you are performing `noxconditional` estimation; [SEM] `sem option noxconditional`.

b. Latent exogenous variables are assumed to have mean 0. Means of latent variables are not estimated by default. If you specify enough normalization constraints to identify the mean of a latent exogenous variable, you can specify `means(name)` to indicate that the mean should be estimated in either.

c. Endogenous variables have no separate mean. Their means are those implied by the model. The `means()` option may not be used with endogenous variables.

d. Error variables have mean 0 and this cannot be modified. The `means()` option may not be used with error variables.

To constrain the mean to a fixed value, such as 57, code

\[
\ldots, \ldots \text{means(name@57)}
\]

Separate `means()` options may be combined:

\[
\ldots, \ldots \text{means(name@57 name2@100)}
\]

11. Fixed-value constraints may be specified for a path, variance, covariance, or mean by using `@` (the “at” symbol). For example,

\[
(\text{name1 <- name2@1})
\]

\[
(\text{name1 <- name2@1 name3@1})
\]

\[
\ldots, \ldots \text{var(name@100)}
\]

\[
\ldots, \ldots \text{cov(name1*name2@223)}
\]

\[
\ldots, \ldots \text{cov(name1@1 name2@1 name1*name2@.8)}
\]

\[
\ldots, \ldots \text{means(name@57)}
\]
12. Symbolic constraints may be specified for a path, variance, covariance, or mean by using @ (the “at” symbol). For example,

\[(\text{name1} \leftarrow \text{name2}@c1) \ (\text{name3} \leftarrow \text{name4}@c1)\]
\[\ldots, \ldots \ \text{var}(\text{name1}@c1 \ \text{name2}@c1)\]
\[\ldots, \ldots \ \text{cov}(\text{name1}@01 \ \text{name2}@01 \ \text{name3}@01 \ \text{name1}*\text{name2}@c1 \ \text{name1}*\text{name3}@c1)\]
\[\ldots, \ldots \ \text{means}(\text{name1}@c1 \ \text{name2}@c1)\]

\[(\text{name1} \leftarrow \text{name2}@c1) \ldots, \ \text{var}(\text{name3}@c1) \ \text{means}(\text{name4}@c1)\]

Symbolic names are just names from 1 to 32 characters in length. Symbolic constraints constrain equality. For simplicity, all constraints below will have names c1, c2, ....

13. Linear combinations of symbolic constraints may be specified for a path, variance, covariance, or mean by using @ (the “at” symbol). For example,

\[(\text{name1} \leftarrow \text{name2}@c1) \ (\text{name3} \leftarrow \text{name4}(2*c1))\]
\[\ldots, \ldots \ \text{var}(\text{name1}@c1 \ \text{name2}(c1/2))\]
\[\ldots, \ldots \ \text{cov}(\text{name1}@01 \ \text{name2}@01 \ \text{name3}@01 \ \text{name1}*\text{name2}@c1 \ \text{name1}*\text{name2}@0(c1/2))\]
\[\ldots, \ldots \ \text{means}(\text{name1}@0c1 \ \text{name2}@0(3*c1+10))\]
\[(\text{name1} \leftarrow \text{name2}(c1/2)) \ldots, \ \text{var}(\text{name3}@c1) \ \text{means}(\text{name4}@0(2*c1))\]

14. All equations in the model are assumed to have an intercept (to include observed exogenous variable _cons) unless the noconstant option (abbreviation nocons) is specified, and then all equations are assumed not to have an intercept (not to include _cons). (There are some exceptions to this in gsem because some generalized linear models have no intercept or even the concept of an intercept.)

Regardless of whether noconstant is specified, you may explicitly refer to observed exogenous variable _cons.

The following path specifications are ways of writing the same model:

\[(\text{name1} \leftarrow \text{name2}) \ (\text{name1} \leftarrow \text{name3})\]
\[(\text{name1} \leftarrow \text{name2}) \ (\text{name1} \leftarrow \text{name3}) \ (\text{name1} \leftarrow _\text{cons})\]
\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3}\]
\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3} \ _\text{cons}\]

There is no reason to explicitly specify _cons unless you have also specified the noconstant option and want to include _cons in some equations but not others, or regardless of whether you specified the noconstant option, you want to place a constraint on its path coefficient. For example,

\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3} \ _\text{cons}@c1) \ (\text{name4} \leftarrow \text{name5}) \ _\text{cons}@c1)\]

15. The noconstant option may be specified globally or within a path specification. That is,

\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3}) \ (\text{name4} \leftarrow \text{name5}), \ \text{nocons}\]
suppresses the intercepts in both equations. Alternatively,

\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3}, \ \text{nocons}) \ (\text{name4} \leftarrow \text{name5})\]
suppresses the intercept in the first equation but not the second, whereas

\[(\text{name1} \leftarrow \text{name2}) \ _\text{name3}) \ (\text{name4} \leftarrow \text{name5}, \ \text{nocons})\]
suppresses the intercept in the second equation but not the first.
In addition, consider the equation

\[(\text{name1} \leftarrow \text{name2} \text{name3}, \text{nocons})\]

This can be written equivalently as

\[(\text{name1} \leftarrow \text{name2}, \text{nocons}) (\text{name1} \leftarrow \text{name3}, \text{nocons})\]

16. Initial values (starting values) may be specified for a path, variance, covariance, or mean by using the \text{init(\#)} suboption:

\[(\text{name1} \leftarrow (\text{name2}, \text{init}(0)))\]
\[(\text{name1} \leftarrow (\text{name2}, \text{init}(0)) \text{name3})\]
\[(\text{name1} \leftarrow (\text{name2}, \text{init}(0)) (\text{name3}, \text{init}(5)))\]
\[\ldots, \ldots \text{var}((\text{name3}, \text{init}(1)))\]
\[\ldots, \ldots \text{cov}((\text{name4*name5}, \text{init}.5)))\]
\[\ldots, \ldots \text{means}((\text{name5}, \text{init}(0)))\]

The initial values may be combined with symbolic constraints:

\[(\text{name1} \leftarrow (\text{name2@c1}, \text{init}(0)))\]
\[(\text{name1} \leftarrow (\text{name2@c1}, \text{init}(0)) \text{name3})\]
\[(\text{name1} \leftarrow (\text{name2@c1}, \text{init}(0)) (\text{name3@c2}, \text{init}(5)))\]
\[\ldots, \ldots \text{var}((\text{name3@c1}, \text{init}(1)))\]
\[\ldots, \ldots \text{cov}((\text{name4*name5@c1}, \text{init}.5)))\]
\[\ldots, \ldots \text{means}((\text{name5@c1}, \text{init}(0)))\]

Also see

[SEM] \text{sem} — Structural equation model estimation command
[SEM] \text{gsem} — Generalized structural equation model estimation command
[SEM] \text{sem path notation extensions} — Command syntax for path diagrams
[SEM] \text{gsem path notation extensions} — Command syntax for path diagrams
[SEM] \text{Intro 2} — Learning the language: Path diagrams and command language
[SEM] \text{Intro 6} — Comparing groups
sem and gsem syntax options — Options affecting interpretation of syntax

Description

These options affect some minor issues of how sem and gsem interpret what you type.

Syntax

sem paths ..., ... syntax_options
gsem paths ..., ... syntax_options

<table>
<thead>
<tr>
<th>syntax_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>latent(names)</td>
<td>explicitly specify latent variable names</td>
</tr>
<tr>
<td>nocapslatent</td>
<td>do not treat capitalized Names as latent</td>
</tr>
</tbody>
</table>

where names is a space-separated list of the names of the latent variables.

Options

latent(names) specifies that names is the full set of names of the latent variables. sem and gsem ordinarily assume that latent variables have the first letter capitalized and observed variables have the first letter lowercased; see [SEM] sem and gsem path notation. When you specify latent(names), sem and gsem treat the listed variables as the latent variables and all other variables, regardless of capitalization, as observed. latent() implies nocapslatent.

nocapslatent specifies that having the first letter capitalized does not designate a latent variable. This option can be used when fitting models with observed variables only where some observed variables in the dataset have the first letter capitalized.

Remarks and examples

We recommend using the default naming convention. If your dataset contains variables with the first letter capitalized, it is easy to convert the variables to have lowercase names by typing

```
    . rename *, lower
```

See [D] rename group.

Also see

[SEM] sem — Structural equation model estimation command
[SEM] gsem — Generalized structural equation model estimation command
[SEM] sem and gsem path notation — Command syntax for path diagrams
**Description**

These options control how results are obtained.

**Syntax**

```
sem paths ..., ... estimation_options
```

<table>
<thead>
<tr>
<th>estimation_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>method(method)</code></td>
<td><code>method</code> may be <code>ml</code>, <code>mlmv</code>, or <code>adf</code></td>
</tr>
<tr>
<td><code>vce(vcetype)</code></td>
<td><code>vcetype</code> may be <code>oim</code>, <code>eim</code>, <code>opg</code>, <code>sbentler</code>, <code>robust</code>, <code>cluster clustvar</code>, <code>bootstrap</code>, or <code>jackknife</code></td>
</tr>
<tr>
<td><code>nm1</code></td>
<td>compute sample variance rather than ML variance</td>
</tr>
<tr>
<td><code>noxconditional</code></td>
<td>compute covariances, etc., of observed exogenous variables</td>
</tr>
<tr>
<td><code>allmissing</code></td>
<td>for use with <code>method(mlmv)</code></td>
</tr>
<tr>
<td><code>noivstart</code></td>
<td>skip calculation of starting values</td>
</tr>
<tr>
<td><code>noestimate</code></td>
<td>do not fit the model; instead show starting values</td>
</tr>
</tbody>
</table>

**Options**

`method()` and `vce()` specify the method used to obtain parameter estimates and the technique used to obtain the variance–covariance matrix of the estimates. See [SEM] sem option method().

`nm1` specifies that the variances and covariances used in the SEM equations be the sample variances (divided by \( N - 1 \)) and not the asymptotic variances (divided by \( N \)). This is a minor technical issue of little importance unless you are trying to match results from other software that assumes sample variances. `SEM` assumes asymptotic variances.

`noxconditional` states that you wish to include the means, variances, and covariances of the observed exogenous variables among the parameters to be estimated by `sem`. See [SEM] sem option noxconditional.

`allmissing` specifies how missing values should be treated when `method(mlmv)` is also specified.

Usually, `SEM` omits from the estimation sample observations that contain missing values of any of the observed variables used in the model. `method(mlmv)`, however, can deal with these missing values, and in that case, observations containing missing are not omitted.

Even so, `SEM`, `method(mlmv)` does omit observations containing `.a`, `.b`, `.c`, `.z` from the estimation sample. `SEM` assumes you do not want these observations used because the missing value is not missing at random. If you want `SEM` to include these observations in the estimation sample, specify the `allmissing` option.
noivstart is an arcane option that is of most use to programmers. It specifies that \texttt{sem} is to skip efforts to produce good starting values with instrumental-variable techniques, techniques that require computer time. If you specify this option, you should specify all the starting values. Any starting values not specified will be assumed to be 0 (means, path coefficients, and covariances) or some simple function of the data (variances).

\texttt{noestimate} specifies that the model is not to be fit. Instead, starting values are to be shown and they are to be shown using the \texttt{coeflegend} style of output. An important use of this is to improve starting values when your model is having difficulty converging. You can do the following:

\begin{verbatim}
 . sem ..., ... noestimate
 . matrix b = e(b)
 . ... (modify elements of b) ...
 . sem ..., ... from(b)
\end{verbatim}

\texttt{maximize_options} specify the standard and rarely specified options for controlling the maximization process for \texttt{sem}; see [\texttt{R}] \texttt{Maximize}. The relevant options for \texttt{sem} are \texttt{difficult}, \texttt{technique(algorithm_spec)}, \texttt{iterate(#)}, \texttt{[no]log}, \texttt{trace}, \texttt{gradient}, \texttt{showstep}, \texttt{hessian}, \texttt{tolerance(#)}, \texttt{ltolerance(#)}, and \texttt{nrtolerance(#)}.

\texttt{satopts(maximize_options)} is a rarely specified option and is only relevant if you specify the \texttt{method(mlmv)} option. \texttt{sem} reports a test for model versus saturated at the bottom of the output. Thus \texttt{sem} needs to obtain the saturated fit. In the case of \texttt{method(ml)} or \texttt{method(adf)}, \texttt{sem} can make a direct calculation. In the other case of \texttt{method(mlmv)}, \texttt{sem} must actually fit the saturated model. The maximization options specified inside \texttt{satopts()} control that maximization process. It is rare that you need to specify the \texttt{satopts()} option, even if you find it necessary to specify the overall \texttt{maximize_options}.

\texttt{baseopts(maximize_options)} is a rarely specified option and an irrelevant one unless you also specify \texttt{method(mlmv)} or \texttt{method(adf)}. When fitting the model, \texttt{sem} records information about the baseline model for later use by \texttt{estat gof}, should you use that command. Thus \texttt{sem} needs to obtain the baseline fit. In the case of \texttt{method(ml)}, \texttt{sem} can make a direct calculation. In the cases of \texttt{method(mlmv)} and \texttt{method(adf)}, \texttt{sem} must actually fit the baseline model. The maximization options specified inside \texttt{baseopts()} control that maximization process. It is rare that you need to specify the \texttt{baseopts()} option even if you find it necessary to specify the overall \texttt{maximize_options}.

Remarks and examples

The most commonly specified option among this group is \texttt{vce()}. See [\texttt{SEM}] \texttt{sem option method()}, [\texttt{SEM}] \texttt{Intro 8}, and [\texttt{SEM}] \texttt{Intro 9}.

Also see

[\texttt{SEM}] \texttt{sem} — Structural equation model estimation command

[\texttt{SEM}] \texttt{sem option method()} — Specifying method and calculation of VCE

[\texttt{SEM}] \texttt{sem option noxconditional} — Computing means, etc., of observed exogenous variables

[\texttt{SEM}] \texttt{Intro 8} — Robust and clustered standard errors

[\texttt{SEM}] \texttt{Intro 9} — Standard errors, the full story
**sem group options — Fitting models on different groups**

### Description

`sem` can fit combined models across subgroups of the data while allowing some parameters to vary and constraining others to be equal across subgroups. These subgroups could be males and females, age category, and the like.

`sem` performs such estimation when the `group(varname)` option is specified. The `ginvariant(pclassname)` option specifies which parameters are to be constrained to be equal across the groups.

### Syntax

```
sem paths ... , ... group_options
```

<table>
<thead>
<tr>
<th>group_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>group(varname)</code></td>
<td>fit model for different groups</td>
</tr>
<tr>
<td><code>ginvariant(pclassname)</code></td>
<td>specify parameters that are equal across groups</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pclassname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>scoef</code></td>
<td>structural coefficients</td>
</tr>
<tr>
<td><code>scons</code></td>
<td>structural intercepts</td>
</tr>
<tr>
<td><code>mcoef</code></td>
<td>measurement coefficients</td>
</tr>
<tr>
<td><code>mcons</code></td>
<td>measurement intercepts</td>
</tr>
<tr>
<td><code>serrvar</code></td>
<td>covariances of structural errors</td>
</tr>
<tr>
<td><code>merrvar</code></td>
<td>covariances of measurement errors</td>
</tr>
<tr>
<td><code>smerrcov</code></td>
<td>covariances between structural and measurement errors</td>
</tr>
<tr>
<td><code>meanex</code></td>
<td>means of exogenous variables</td>
</tr>
<tr>
<td><code>covex</code></td>
<td>covariances of exogenous variables</td>
</tr>
<tr>
<td><code>all</code></td>
<td>all the above</td>
</tr>
<tr>
<td><code>none</code></td>
<td>none of the above</td>
</tr>
</tbody>
</table>

`ginvariant(mcoef mcons)` is the default if `ginvariant()` is not specified.

`meanex`, `covex`, and `all` exclude the observed exogenous variables (that is, they include only the latent exogenous variables) unless you specify the `noxconditional` option or the `noxconditional` option is otherwise implied; see [SEM] `sem option noxconditional`. This is what you would desire in most cases.
**Options**

`group(varname)` specifies that the model be fit as described above. `varname` specifies the name of a numeric variable that records the group to which the observation belongs.

If you are using summary statistics data in place of raw data, `varname` is the name of the group variable as reported by `ssd describe`; see [SEM] `ssd`.

`ginvariant(pclassname)` specifies which classes of parameters of the model are to be constrained to be equal across groups. The classes are defined above. The default is `ginvariant(mcoef mcons)` if the option is not specified.

**Remarks and examples**

See [SEM] `Intro 6`, and see [SEM] `Example 20` and [SEM] `Example 23`.

**Also see**

[SEM] `sem` — Structural equation model estimation command

[SEM] `Intro 6` — Comparing groups

[SEM] `Example 20` — Two-factor measurement model by group

[SEM] `Example 23` — Specifying parameter constraints across groups
**Description**

*paths* and the options above describe the model to be fit by *sem*.

**Syntax**

```
sem paths ..., ... model_description_options
```

---

**Description**

* covariance() path notation for treatment of covariances; see [SEM] sem and gsem path notation*

* variance() path notation for treatment of variances; see [SEM] sem and gsem path notation*

* means() path notation for treatment of means; see [SEM] sem and gsem path notation*

* covstructure() alternative method to place restrictions on covariances; see [SEM] sem and gsem option covstructure()*

noconstant do not fit intercepts

nomeans do not fit means or intercepts

noanchor do not apply default anchoring

forcenoanchor programmer’s option

* reliability() reliability of measurement variables; see [SEM] sem and gsem option reliability()*

constraints() specify constraints; see [SEM] sem and gsem option constraints()*

from() specify starting values; see [SEM] sem and gsem option from()*

* Option may be specified more than once.

**Options**

* covariance(), variance(), and means() fully describe the model to be fit. See [SEM] sem and gsem path notation.*

* covstructure() provides a convenient way to constrain covariances in your model. Alternatively or in combination, you can place constraints by using the standard path notation. See [SEM] sem and gsem option covstructure().*

* noconstant specifies that all intercepts be constrained to 0. See [SEM] sem and gsem path notation.*

* nomeans specifies that means and intercepts not be fit. The means and intercepts are concentrated out of the function being optimized, which is typically the likelihood function. Results for all other parameters are the same whether or not this option is specified.
This option is seldom specified. `sem` issues this option to itself when you use summary statistics data that do not include summary statistics for the means.

`noanchor` specifies that `sem` is not to check for lack of identification and fill in anchors where needed. `sem` is instead to issue an error message if anchors would be needed. You specify this option when you believe you have specified the necessary normalization constraints and want to hear about it if you are wrong. See Identification 2: Normalization constraints (anchoring) in [SEM] Intro 4.

`forcenoanchor` is similar to `noanchor` except that rather than issue an error message, `sem` proceeds to estimation. There is no reason you should specify this option. `forcenoanchor` is used in testing of `sem` at StataCorp.

`reliability()` specifies the fraction of variance not due to measurement error for a variable. See [SEM] sem and gsem option reliability().

`constraints()` specifies parameter constraints you wish to impose on your model; see [SEM] sem and gsem option constraints(). Constraints can also be specified as described in [SEM] sem and gsem path notation, and they are usually more conveniently specified using the path notation.

`from()` specifies the starting values to be used in the optimization process; see [SEM] sem and gsem option from(). Starting values can also be specified by using the `init()` suboption as described in [SEM] sem and gsem path notation.

Remarks and examples

To use `sem` successfully, you need to understand `paths`, `covariance()`, `variance()`, and `means()`; see Using path diagrams to specify standard linear SEMs in [SEM] Intro 2 and [SEM] sem and gsem path notation.

`covstructure()` is often convenient; see [SEM] sem and gsem option covstructure().

Also see

[SEM] sem — Structural equation model estimation command
[SEM] Intro 2 — Learning the language: Path diagrams and command language
[SEM] sem and gsem path notation — Command syntax for path diagrams
[SEM] sem and gsem option covstructure() — Specifying covariance restrictions
[SEM] sem and gsem option reliability() — Fraction of variance not due to measurement error
[SEM] sem and gsem option constraints() — Specifying constraints
[SEM] sem and gsem option from() — Specifying starting values
sem option method() — Specifying method and calculation of VCE

Description

sem option method() specifies the method used to obtain the estimated parameters.

sem option vce() specifies the technique used to obtain the variance–covariance matrix of the estimates (VCE), which includes the reported standard errors.

Syntax

```
sem ... [ , ... method(method) vce(vcetype) ... ]
```

**method**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ml</td>
</tr>
<tr>
<td>maximum likelihood; the default</td>
</tr>
<tr>
<td>mlmv</td>
</tr>
<tr>
<td>ml with missing values</td>
</tr>
<tr>
<td>adf</td>
</tr>
<tr>
<td>asymptotic distribution free</td>
</tr>
</tbody>
</table>

**vcetype**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oim</td>
</tr>
<tr>
<td>observed information matrix; the default</td>
</tr>
<tr>
<td>eim</td>
</tr>
<tr>
<td>expected information matrix</td>
</tr>
<tr>
<td>opg</td>
</tr>
<tr>
<td>outer product of gradients</td>
</tr>
<tr>
<td>sbentler</td>
</tr>
<tr>
<td>Satorra–Bentler estimator</td>
</tr>
<tr>
<td>robust</td>
</tr>
<tr>
<td>Huber/White/sandwich estimator</td>
</tr>
<tr>
<td>cluster clustvar</td>
</tr>
<tr>
<td>generalized Huber/White/sandwich estimator</td>
</tr>
<tr>
<td>bootstrap [, bootstrap_options]</td>
</tr>
<tr>
<td>bootstrap estimation</td>
</tr>
<tr>
<td>jackknife [, jackknife_options]</td>
</tr>
<tr>
<td>jackknife estimation</td>
</tr>
</tbody>
</table>

pweights and iweights are not allowed with sbentler.

The following combinations of method() and vce() are allowed:

<table>
<thead>
<tr>
<th></th>
<th>oim</th>
<th>eim</th>
<th>opg</th>
<th>sbentler</th>
<th>robust</th>
<th>cluster</th>
<th>bootstrap</th>
<th>jackknife</th>
</tr>
</thead>
<tbody>
<tr>
<td>ml</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>mlmv</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>adf</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
Options

`method(method)` specifies the method used to obtain parameter estimates. `method(ml)` is the default.

`vce(vcetype)` specifies the technique used to obtain the VCE. `vce(oim)` is the default.

Remarks and examples

See [SEM] Intro 4, [SEM] Intro 8, and [SEM] Intro 9.

Also see

[SEM] sem — Structural equation model estimation command
[SEM] Intro 4 — Substantive concepts
[SEM] Intro 8 — Robust and clustered standard errors
[SEM] Intro 9 — Standard errors, the full story
[SEM] Example 26 — Fitting a model with data missing at random
Description

sem has a noxconditional option that you may rarely wish to specify. The option is described below.

Syntax

sem ... [, ... noxconditional ... ]

Option

noxconditional states that you wish to include the means, variances, and covariances of the observed exogenous variables among the parameters to be estimated by sem.

Remarks and examples

Remarks are presented under the following headings:

What is x conditional?
When to specify noxconditional
Option forcexconditional (a technical note)

What is x conditional?

In many cases, sem does not include the means, variances, and covariances of observed exogenous variables among the parameters to be estimated. When sem omits them, the estimator of the model is said to be x conditional. Rather than estimating the values of the means, variances, and covariances, sem uses the separately calculated observed values of those statistics. sem does this to save time and memory.

sem does not use the x-conditional calculation when it would be inappropriate.

The noxconditional option prevents sem from using the x-conditional calculation. You specify noxconditional on the sem command:

. sem ..., ... noxconditional

Do not confuse the x-conditional calculation with the assumption of conditional normality discussed in [SEM] Intro 4. The x-conditional calculation is appropriate even when the assumption of conditional normality is inappropriate.
When to specify noxconditional

It is never inappropriate to specify the noxconditional option. Be aware, however:

1. If you are using the default method(ml), estimated point estimates and standard errors will be the same.

2. If you are using method(adf), estimated point estimates and standard errors will be slightly different, asymptotically equivalent, and there is no reason to prefer one set of estimates over the other.

3. If you are using method(mlmv), the situation is the same as in statement 1.

4. Regardless of the estimation method used, calculation of results will require more computer time and memory. The memory requirements increase quadratically with the total number of estimated parameters in your model. If you have $k_1$ observed exogenous variables and $k_2$ latent exogenous variables, the number of added parameters from noxconditional is $k_1 + k_1(k_1 + 1)/2 + k_1 k_2$. The resulting total memory requirements can be so great as to require more memory than your computer can provide.

To make statements 1–4 true, there are two cases when sem specifies noxconditional for you:

1. sem defaults to noxconditional whenever you constrain a mean, variance, or covariance of an observed exogenous variable. For example,

   . sem ..., ... means(x1@m x2@m)
   . sem ..., ... var(x1@v x2@v)
   . sem ..., ... cov(x1*x2@c x1*x3@c)
   . sem ..., ... covstruct(_OEx, diagonal)

   See [SEM] sem and gsem path notation and [SEM] sem and gsem option covstructure().

2. sem defaults to noxconditional whenever you use method(mlmv) and there are missing values among the observed exogenous variables.

There are only three reasons for you to specify the noxconditional option:

1. Specify noxconditional if you subsequently wish to test means, variances, or covariances of observed exogenous variables with postestimation commands. For example,

   . sem ..., ... noxconditional
   . sem, coeflegend
   . test _b[/means(x1)] == _b[/means(x2)]

2. Specify noxconditional if you are fitting a model with the group() option.

3. Specify noxconditional if you also specify the ginvariant() option, and you want the ginvariant() classes meanex, covex, or all to include the observed exogenous variables. For example,

   . sem ..., ... by(agegrp) ginvariant(all) noxconditional

You may also wish to specify noxconditional when comparing results with those from other packages. Many packages use the noxconditional approach when using an estimation method other than maximum likelihood (ML). Correspondingly, most packages use the x-conditional calculation when using ML.
Option forceconditional (a technical note)

In addition to noxconditional, sem has a forceconditional option:

\[ \text{sem} \ldots [, \ldots \text{forceconditional} \ldots ] \]

This option turns off sem’s switching away from the x-conditional calculation when that is required. Do not specify this option unless you are exploring the behavior of x-conditional calculation in cases where it is theoretically inappropriate.

Also see

[SEM] sem — Structural equation model estimation command
sem option select() — Using sem with summary statistics data

Description

`sem` may be used with summary statistics data (SSD), data containing only summary statistics such as the means, standard deviations or variances, and correlations and covariances of the underlying, raw data.

You enter SSD with the `ssd` command; see `[SEM] ssd`.

To fit a model with `sem`, there is nothing special you have to do except specify the `select()` option where you would usually specify `if exp`.

Syntax

```plaintext
sem ... [, ... select(# [ # ... ] ) ... ]
```

Option

`select(# [ # ... ] )` is allowed only when you have SSD in memory. It specifies which groups should be used.

Remarks and examples

See `[SEM] Intro 11`.

`sem option select()` is the SSD alternative for `if exp` if you only had the underlying, raw data in memory. With the underlying raw data, where you would usually type

```
. sem ... if agegrp==1 | agegrp==3, ...
```

with SSD in memory, you type

```
. sem ..., ... select(1 3)
```

You may select only groups for which you have separate summary statistics recorded in your summary statistics dataset; the `ssd describe` command will list the group variable, if any. See `[SEM] ssd`.

By the way, `select()` may be combined with `sem option group()`. Where you might usually type

```
. sem ... if agegrp==1 | agegrp==3, ... group(agegrp)
```

you could type

```
. sem ..., ... select(1 3) group(agegrp)
```
The above restricts `sem` to age groups 1 and 3, so the result will be an estimation of a combined model of age groups 1 and 3 with some coefficients allowed to vary between the groups and other coefficients constrained to be equal across the groups. See [SEM] `sem group options`.

Also see

[SEM] `sem` — Structural equation model estimation command

[SEM] `Intro 11` — Fitting models with summary statistics data (sem only)
sem path notation extensions — Command syntax for path diagrams

Description

This entry concerns sem only.

The command syntax for describing your SEMs is fully specified by paths, covariance(), variance(), covstructure(), and means(). How that works is described in [SEM] sem and gsem path notation and [SEM] sem and gsem option covstructure(). See those sections before reading this section.

This entry concerns the path features unique to sem, and that has to do with the group() option for comparing different groups.

Syntax

```
sem paths ... [, covariance() variance() means() group(varname)]
sem paths ... [, covstructure() means() group(varname)]
```

paths specifies the direct paths between the variables of your model.

The model to be fit is fully described by paths, covariance(), variance(), covstructure(), and means().

The syntax of these elements is modified (generalized) when the group() option is specified.

Options

covariance(), variance(), and means() are described in [SEM] sem and gsem path notation.

covstructure() is described in [SEM] sem and gsem option covstructure().

group(varname) allows models specified with paths, covariance(), variance(), covstructure(), and means() to be automatically generalized (interacted) with the groups defined by varname; see [SEM] Intro 6. The syntax of paths and the arguments of covariance(), variance(), covstructure(), and means() gain an extra syntactical piece when group() is specified.

Remarks and examples

The model you wish to fit is fully described by the paths, covariance(), variance(), covstructure(), and means() that you type. The group(varname) option,

```
. sem ..., ... group(varname)
```
specifies that the model be fit separately for the different values of `varname`. `varname` might be `sex` and then the model would be fit separately for males and females, or `varname` might be something else and perhaps take on more than two values.

Whatever `varname` is, `group(varname)` defaults to letting some of the path coefficients, covariances, variances, and means of your model vary across the groups and constraining others to be equal. Which parameters vary and which are constrained is described in [SEM] `sem group options`, but that is a minor detail right now.

In what follows, we will assume that `varname` is `mygrp` and takes on three values. Those values are 1, 2, and 3, but they could just as well be 2, 9, and 12.

Consider typing

```
   . sem ..., ...
```

and typing

```
   . sem ..., ... group(mygrp)
```

Whatever the `paths`, `covariance()`, `variance()`, `covstructure()`, and `means()` are that describe the model, there are now three times as many parameters because each group has its own unique set. In fact, when you give the second command, you are not merely asking for three times the parameters, you are specifying three models, one for each group! In this case, you specified the same model three times without knowing it.

You can vary the model specified across groups.

1. Let’s write the model you wish to fit as

```
   . sem (a) (b) (c), cov(d) cov(e) var(f)
```

where `a`, `b`, ..., `f` stand for what you type. In this generic example, we have two `cov()` options just because multiple `cov()` options often occur in real models. When you type

```
   . sem (a) (b) (c), cov(d) cov(e) var(f) group(mygrp)
```

results are as if you typed

```
   . sem (1: a) (2: a) (3: a)   ///
       (1: b) (2: b) (3: b)   ///
       (1: c) (2: c) (3: c),   ///
       cov(1: d) cov(2: d) cov(3: d) ///
       cov(1: e) cov(2: e) cov(3: e) ///
       var(1: f) cov(2: f) cov(3: f) group(mygrp)
```

The 1:, 2:, and 3: identify the groups for which paths, covariances, or variances are being added, modified, or constrained.

If `mygrp` contained the unique values 5, 8, and 10 instead of 1, 2, and 3, then 5: would appear in place of 1:; 8: would appear in place of 2:; and 10: would appear in place of 3:.

2. Consider the model

```
   . sem (y <- x) (b) (c), cov(d) cov(e) var(f) group(mygrp)
```

If you wanted to constrain the path coefficient `(y <- x)` to be the same across all three groups, you could type

```
   . sem (y <- x@c1) (b) (c), cov(d) cov(e) var(f) group(mygrp)
```

See item 12 in [SEM] `sem and gsem path notation` for more examples of specifying constraints. This works because the expansion of `(y <- x@c1)` is

```
   (1: y <- x@c1) (2: y <- x@c1) (3: y <- x@c1)
```
3. Consider the model

\[
\text{. sem (y <- x) (b) (c), cov(d) cov(e) var(f) group(mygrp)}
\]

If you wanted to constrain the path coefficient \((y \leftarrow x)\) to be the same in groups 2 and 3, you could type

\[
\text{. sem (1: y <- x) (2: y <- x@c1) (3: y <- x@c1) (b) (c), } \quad /\!\!/ \quad \text{cov(d) cov(e) var(f) group(mygrp)}
\]

4. Instead of following item 3, you could type

\[
\text{. sem (y <- x) (2: y <- x@c1) (3: y <- x@c1) (b) (c), } \quad /\!\!/ \quad \text{cov(d) cov(e) var(f) group(mygrp)}
\]

The part \((y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)\) expands to

\[
(1: y \leftarrow x) (2: y \leftarrow x) (3: y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)
\]

and thus the path is defined twice for group 2 and twice for group 3. When a path is defined more than once, the definitions are combined. In this case, the second definition adds more information, so the result is as if you typed

\[
(1: y \leftarrow x) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)
\]

5. Instead of following item 3 or item 4, you could type

\[
\text{. sem (y <- x@c1) (1: y <- x@c2) (b) (c), } \quad /\!\!/ \quad \text{cov(d) cov(e) var(f) group(mygrp)}
\]

The part \((y \leftarrow x@c1) (1: y \leftarrow x@c2)\) expands to

\[
(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (1: y \leftarrow x@c2)
\]

When results are combined from repeated definitions, then definitions that appear later take precedence. In this case, results are as if the expansion read

\[
(1: y \leftarrow x@c2) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)
\]

Thus coefficients for groups 2 and 3 are constrained. The group-1 coefficient is constrained to \(c2\). If \(c2\) appears nowhere else in the model specification, then results are as if the path for group 1 were unconstrained.

6. Instead of following item 3, item 4, or item 5, you could not type

\[
\text{. sem (y <- x@c1) (1: y <- x) (b) (c), } \quad /\!\!/ \quad \text{cov(d) cov(e) var(f) group(mygrp)}
\]

The expansion of \((y \leftarrow x@c1) (1: y \leftarrow x)\) reads

\[
(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1) (1: y \leftarrow x)
\]

and you might think that \(1: y \leftarrow x\) would replace \(1: y \leftarrow x@c1\). Information, however, is combined, and even though precedence is given to information appearing later, silence does not count as information. Thus the expanded and reduced specification reads the same as if \(1: y \leftarrow x\) was never specified:

\[
(1: y \leftarrow x@c1) (2: y \leftarrow x@c1) (3: y \leftarrow x@c1)
\]

7. Items 1–6, stated in terms of \textit{paths}, apply equally to what is typed inside the \texttt{means()}, \texttt{variance()}, \texttt{covariance()}, and \texttt{covstructure()} options. For instance, if you typed

\[
\text{. sem (a) (b) (c), var(e.y@c1) group(mygrp)}
\]
then you are constraining the variance to be equal across all three groups.

If you wanted to constrain the variance to be equal in groups 2 and 3, you could type

```
    . sem (a) (b) (c), var(e.y) var(2: e.y@c1) var(3: e.y@c1), group(mygrp)
```

You could omit typing `var(e.y)` because it is implied. Alternatively, you could type

```
    . sem (a) (b) (c), var(e.y@c1) var(1: e.y@c2) group(mygrp)
```

You could not type

```
    . sem (a) (b) (c), var(e.y@c1) var(1: e.y) group(mygrp)
```

because silence does not count as information when specifications are combined.

Similarly, if you typed

```
    . sem (a) (b) (c), cov(e.y1*e.y2@c1) group(mygrp)
```

then you are constraining the covariance to be equal across all groups. If you wanted to constrain the covariance to be equal in groups 2 and 3, you could type

```
    . sem (a) (b) (c), cov(e.y1*e.y2) ///
      cov(2: e.y1*e.y2@c1) cov(3: e.y1*e.y2@c1) ///
      group(mygrp)
```

You could not omit `cov(e.y1*e.y2)` because it is not assumed. By default, error variables are assumed to be uncorrelated. Omitting the option would constrain the covariance to be 0 in group 1 and to be equal in groups 2 and 3.

Alternatively, you could type

```
    . sem (a) (b) (c), cov(e.y1*e.y2@c1) ///
      cov(1: e.y1*e.y2@c2) ///
      group(mygrp)
```

8. In the examples above, we have referred to the groups with their numeric values, 1, 2, and 3. Had the values been 5, 8, and 10, then we would have used those values.

If the group variable `mygrp` has a value label, you can use the label to refer to the group. For instance, imagine `mygrp` is labeled as follows:

```
    . label define grpvals 1 Male 2 Female 3 "Unknown sex"
    . label values mygrp grpvals
```

We could type

```
    . sem (y <- x) (Female: y <- x@c1) (Unknown sex: y <- x@c1) ..., ...
```

or we could type

```
    . sem (y <- x) (2: y <- x@c1) (3: y <- x@c1) ..., ...
```

Also see

[SEM] `sem` — Structural equation model estimation command

[SEM] `sem and gsem path notation` — Command syntax for path diagrams

[SEM] `Intro 2` — Learning the language: Path diagrams and command language

[SEM] `Intro 6` — Comparing groups
Postestimation commands

The following are the postestimation commands that you can use after estimation by `sem`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>estat framework</td>
<td>display results in modeling framework (matrix form)</td>
</tr>
<tr>
<td>estat gof</td>
<td>overall goodness of fit</td>
</tr>
<tr>
<td>estat ggof</td>
<td>group-level goodness of fit</td>
</tr>
<tr>
<td>estat eqgof</td>
<td>equation-level goodness of fit</td>
</tr>
<tr>
<td>estat residuals</td>
<td>matrices of residuals</td>
</tr>
<tr>
<td>estat ic</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>*hausman</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>estat mindices</td>
<td>modification indices (score tests)</td>
</tr>
<tr>
<td>estat scoretests</td>
<td>score tests</td>
</tr>
<tr>
<td>estat ginvariant</td>
<td>test of invariance of parameters across groups</td>
</tr>
<tr>
<td>estat eqtest</td>
<td>equation-level Wald tests</td>
</tr>
<tr>
<td>*lrtest</td>
<td>likelihood-ratio tests</td>
</tr>
<tr>
<td>test</td>
<td>Wald tests</td>
</tr>
<tr>
<td>lincom</td>
<td>linear combination of parameters</td>
</tr>
<tr>
<td>nlcom</td>
<td>nonlinear combination of parameters</td>
</tr>
<tr>
<td>testnl</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
<tr>
<td>estat stdize:</td>
<td>test standardized parameters</td>
</tr>
<tr>
<td>estat teffects</td>
<td>decomposition of effects</td>
</tr>
<tr>
<td>estat stable</td>
<td>assess stability of nonrecursive systems</td>
</tr>
<tr>
<td>estat summarize</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>estat vce</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>predict</td>
<td>factor scores, predicted values, etc.</td>
</tr>
<tr>
<td>margins</td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td>marginsplot</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>estimates</td>
<td>cataloging estimation results</td>
</tr>
</tbody>
</table>

* hausman and lrtest are not appropriate with svy estimation results.

For a summary of postestimation features, see [SEM] Intro 7.

Postestimation commands such `lincom` and `nlcom` require referencing estimated parameter values, which are accessible via `_b[name]`. To find out what the names are, type `sem, coeflegend`. 
margins

Description for margins

margins estimates margins of response for linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options ]
margins [marginlist], predict(statistic ...) [ options ]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>linear predictions for all OEn variables</td>
</tr>
<tr>
<td>xb(varname)</td>
<td>linear prediction for specified OEn variable</td>
</tr>
<tr>
<td>xb</td>
<td>not syntactically compatible with margins</td>
</tr>
<tr>
<td>xblatent</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>xblatent(varlist)</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>latent</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>latent(varlist)</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Key: OEn = observed endogenous

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.

Remarks and examples

This manual entry concerns sem. For information on postestimation features available after gsem, see [SEM] gsem postestimation.

Reference


Also see

[SEM] sem reporting options — Options affecting reporting of results
sem reporting options — Options affecting reporting of results

## Description

These options control how `sem` displays estimation results.

## Syntax

```
sem paths ..., ... reporting_options

sem, reporting_options
```

### reporting_options

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
<th>Remarks and examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>level(#)</code></td>
<td><code>level(95)</code></td>
<td>default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>standardized</code></td>
<td></td>
<td>display standardized coefficients and values</td>
</tr>
<tr>
<td><code>coeflegend</code></td>
<td></td>
<td>display coefficient legend</td>
</tr>
<tr>
<td><code>nocnsreport</code></td>
<td></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><code>nodescr</code></td>
<td><code>nodescr</code></td>
<td>do not display variable classification table</td>
</tr>
<tr>
<td><code>noheader</code></td>
<td></td>
<td>do not display header above parameter table</td>
</tr>
<tr>
<td><code>nofootnote</code></td>
<td><code>nofootnote</code></td>
<td>do not display footnotes below parameter table</td>
</tr>
<tr>
<td><code>notable</code></td>
<td><code>notable</code></td>
<td>do not display parameter table</td>
</tr>
<tr>
<td><code>nofvlabel</code></td>
<td><code>nofvlabel</code></td>
<td>display group values rather than value labels</td>
</tr>
<tr>
<td><code>fvwrap(#)</code></td>
<td><code>fvwrap(#)</code></td>
<td>allow # lines when wrapping long value labels</td>
</tr>
<tr>
<td><code>fvwrapon(style)</code></td>
<td>apply style for wrapping long value labels; style may be word or width</td>
<td></td>
</tr>
<tr>
<td><code>byparm</code></td>
<td></td>
<td>display results in a single table with rows arranged by parameter</td>
</tr>
<tr>
<td><code>showginvariant</code></td>
<td></td>
<td>report all estimated parameters</td>
</tr>
</tbody>
</table>

## Options

### `level(#)`; see [R] Estimation options.

`standardized` displays standardized values, that is, “beta” values for coefficients, correlations for covariances, and 1s for variances. Standardized values are obtained using model-fitted variances (Bollen 1989, 124–125). We recommend caution in the interpretation of standardized values, especially with multiple groups.

`coeflegend` displays the legend that reveals how to specify estimated coefficients in `_b[]` notation, which you are sometimes required to use when specifying postestimation commands.

`nocnsreport` suppresses the display of the constraints. Fixed-to-zero constraints that are automatically set by `sem` are not shown in the report to keep the output manageable.
nodescribe suppresses display of the variable classification table.
noheader suppresses the header above the parameter table, the display that reports the final log-
likelihood value, number of observations, etc.
nofootnote suppresses the footnotes displayed below the parameter table.
notable suppresses the parameter table.
nofvlabel displays group values rather than value labels.
fvwrap(#) specifies how many lines to allow when long value labels must be wrapped. Labels
requiring more than # lines are truncated. This option overrides the fvwrap setting; see [R] set
showbaselevels.

fvwrapon(style) specifies whether value labels that wrap will break at word boundaries or break
based on available space.
fvwrapon(word), the default, specifies that value labels break at word boundaries.
fvwrapon(width) specifies that value labels break based on available space.

This option overrides the fvwrapon setting; see [R] set showbaselevels.
byparm specifies that estimation results with multiple groups be reported in a single table with rows
arranged by parameter. The default is to report results in separate tables for each group.
showginvariant specifies that each estimated parameter be reported in the parameter table. The
default is to report each invariant parameter only once. This option is only effective with the
byparm option.

Remarks and examples

Any of the above options may be specified when you fit the model or when you redisplay results,
which you do by specifying nothing but options after the sem command:

```
. sem (...) (...) ...
(original output displayed)
. sem
(output redisplayed)
. sem, standardized
(standardized output displayed)
. sem, coeflegend
(coefficient-name table displayed)
. sem
(output redisplayed)
```

Reference


Also see

[SEM] sem — Structural equation model estimation command
[SEM] Example 8 — Testing that coefficients are equal, and constraining them
[SEM] Example 16 — Correlation
sem ssd options — Options for use with summary statistics data

Description

Data are sometimes available in summary statistics form only. These summary statistics include means, standard deviations or variances, and correlations or covariances. These summary statistics can be used by sem in place of the underlying raw data.

Syntax

```
sem paths ..., ... ssd_options
```

<table>
<thead>
<tr>
<th>ssd_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>select()</td>
<td>alternative to if exp for SSD</td>
</tr>
<tr>
<td>forcecorrelations</td>
<td>allow groups and pooling of SSD correlations</td>
</tr>
</tbody>
</table>

Options

select() is an alternative to if exp when you are using summary statistics data (SSD). Where you might usually type

```
. sem ... if agegrp==1 | agegrp==3 | agegrp==5, ...
```

with SSD in memory, you type

```
. sem ..., ..., select(1 3 5)
```

See [SEM] sem option select() and [SEM] Intro 11.

forcecorrelations tells sem that it may make calculations that would usually be considered suspicious with SSD that contain only a subset of means, variances (standard deviations), and covariances (correlations). Do not specify this option unless you appreciate the statistical issues that we are about to discuss. There are two cases where forcecorrelations is relevant.

In the first case, sem is unwilling to produce group() estimates if one or more (usually all) of the groups have correlations only defined. You can override that by specifying forcecorrelations, and sem will assume unit variances for the group or groups that have correlations only. Doing this is suspect unless you make ginvariant() all parameters that are dependent on covariances or unless you truly know that the variances are indeed 1.

In the second case, sem is unwilling to pool across groups unless you have provided means and covariances (or means and correlations and standard deviations or variances). Without that information, should the need for pooling arise, sem issues an error message. The forcecorrelations option specifies that sem ignore its rule and pool correlation matrices, treating correlations as if they were covariances when variances are not defined and treating means as if they were 0 when means are not defined. The only justification for making the calculation in this way is that variances truly are 1 and means truly are 0.
Understand that there is nothing wrong with using pure correlation data, or covariance data without the means, so long as you fit models for individual groups. Doing anything across groups basically requires that `sem` have the covariance and mean information.

Remarks and examples

See [SEM Intro 11].

Also see

- [SEM] `sem` — Structural equation model estimation command
- [SEM Intro 11] — Fitting models with summary statistics data (sem only)
- [SEM] `ssd` — Making summary statistics data (sem only)
**Description**

`ssd` allows you to enter SSD to fit SEMs and allows you to create SSD from original, raw data to publish or to send to others (and thus preserve participant confidentiality). Data created by `ssd` may be used with `sem` but not `gsem`.

**Syntax**

To enter summary statistics data (SSD), the commands are

```bash
ssd init varlist
ssd set [#] observations #
ssd set [#] means vector
ssd set [#] {variances|sd} vector
ssd set [#] {covariances|correlations} matrix
ssd addgroup varname (to add second group)
ssd addgroup (to add subsequent groups)
ssd unaddgroup # (to remove last group)
ssd status [#] (to review status)
```

To build SSD from raw data, the command is

```bash
ssd build varlist [, group(varname) clear]
```

To review the contents of SSD, the commands are

```bash
ssd status [#]
ssd describe
ssd list [#]
```
In an emergency (ssd will tell you when), you may use

```
ssd repair
```

In the above,

A *vector* can be any of the following:

1. A space-separated list of numbers, for example,
   ```
   . ssd set means 1 2 3
   ```
2. *(stata)* `matname`
   where `matname` is the name of a Stata $1 \times k$ or $k \times 1$ matrix, for example,
   ```
   . ssd set variances (stata) mymeans
   ```
3. *(mata)* `matname`
   where `matname` is the name of a Mata $1 \times k$ or $k \times 1$ matrix, for example,
   ```
   . ssd set sd (mata) mymeans
   ```

A *matrix* can be any of the following:

1. A space-separated list of numbers corresponding to the rows of the matrix, with backslashes (`\`) between rows. The numbers are either the lower triangle and diagonal or the diagonal and upper triangle of the matrix, for example,
   ```
   . ssd set correlations 1 \ .2 1 \ .3 .5 1
   ```
   or
   ```
   . ssd set correlations 1 .2 .3 \ 1 .5 \ 1
   ```
2. *(ltd)* `# # ...
   which is to say, a space-separated list of numbers corresponding to the lower triangle and diagonal of the matrix, without backslashes between rows, for example,
   ```
   . ssd set correlations (ltd) 1 .2 1 .3 .5 1
   ```
3. *(dut)* `# # ...
   which is to say, a space-separated list of numbers corresponding to the diagonal and upper triangle of the matrix, without backslashes between rows, for example,
   ```
   . ssd set correlations (dut) 1 .2 .3 1 .5 1
   ```
4. *(stata)* `matname`
   where `matname` is the name of a Stata $k \times k$ symmetric matrix, for example,
   ```
   . ssd set correlations (stata) mymat
   ```
5. *(mata)* `matname`
   where `matname` is the name of a Mata $k \times k$ symmetric matrix, for example,
   ```
   . ssd set correlations (mata) mymat
   ```
Options

`group(varname)` is for use with `ssd build`. It specifies that separate groups of summary statistics be produced for each value of `varname`.

`clear` is for use with `ssd build`. It specifies that it is okay to replace the data in memory with SSD even if the original dataset has not been saved since it was last changed.

Remarks and examples

See

[SEM] Intro 11 Fitting models with summary statistics data (sem only)

for an introduction, and see

[SEM] Example 2 Creating a dataset from published covariances
[SEM] Example 19 Creating multiple-group summary statistics data
[SEM] Example 25 Creating summary statistics data from raw data

A summary statistics dataset is different from a regular, raw Stata dataset. Be careful not to use standard Stata data-manipulation commands with SSD in memory. The commands include

`generate`  
`replace`  
`merge`  
`append`  
`drop`  
`sset obs`

to mention a few. You may, however, use `rename` to change the names of the variables.

The other data-manipulation commands can damage your summary statistics dataset. If you make a mistake and do use one of these commands, do not attempt to repair the data yourself. Let `ssd repair` your data by typing

```
.ssd repair
```

`ssd` is usually successful as long as variables or observations have not been dropped.

Every time you use `ssd`, even for something as trivial as describing or listing the data, `ssd` verifies that the data are not corrupted. If `ssd` finds that they are, it suggests you type `ssd repair`:

```
.ssd describe
SSD corrupt
   The summary statistics data should [ssd describes the problem]. The data may be fixable; type ssd repair.
.ssd repair
   (data repaired)
.ssd describe
   (usual output appears)
```

In critical applications, we also recommend you digitally sign your summary statistics dataset:

```
.datasignature set
5:5(65336):3718404259:2275399871 (data signature set)
```
Then at any future time, you can verify the data are still just as they should be:

```
. datasync confirm
(data unchanged since 30jun2018 15:32)
```

The data signature is a function of the variable names. If you rename a variable—something that is allowed—then the data signature will change:

```
. rename varname newname
. datasync confirm
(data have changed since 30jun2018 15:32)
```

In that case, you can re-sign the data:

```
. datasync set, reset
5:5(71728):3718404259:2275399871 (data signature set)
```

Before re-signing, however, if you want to convince yourself that the data are unchanged except for the variable name, type `datasync report`. It is the part of the signature in parentheses that has to do with the variable names. `datasync report` will tell you what the new signature would be, and you can verify that the other components of the signature match.

See [D] `datasync`.

### Stored results

`ssd describe` stores the following in `r()`:

**Scalars**

- `r(N)` number of observations (total across groups)
- `r(k)` number of variables (excluding group variable)
- `r(G)` number of groups
- `r(complete)` 1 if complete, 0 otherwise
- `r(complete_means)` 1 if complete means, 0 otherwise
- `r(complete_covariances)` 1 if complete covariances, 0 otherwise

**Macros**

- `r(v#)` variable names (excluding group variable)
- `r(groupvar)` name of group variable (if there is one)

### Also see

- [SEM] Intro 11 — Fitting models with summary statistics data (sem only)
- [D] `datasync` — Determine whether data have changed
- [SEM] Example 2 — Creating a dataset from published covariances
- [SEM] Example 19 — Creating multiple-group summary statistics data
- [SEM] Example 25 — Creating summary statistics data from raw data
**Description**

test is a postestimation command for use after sem, gsem, and other Stata estimation commands.

test performs the Wald test of the hypothesis or hypotheses that you specify. In the case of sem and gsem, you must use the _b[ ] coefficient notation.

See [R] test. Also documented there is testparm. testparm cannot be used after sem or gsem because its syntax hinges on use of shortcuts for referring to coefficients.

**Menu**

Statistics > SEM (structural equation modeling) > Testing and CIs > Wald tests of linear hypotheses

**Syntax**

```
test coeflist

test exp = exp [ = ... ]

test [eqno] [ : coeflist ]

test [eqno = eqno [ = ... ] ] [ : coeflist ]

test (spec) [ (spec) ... ] [ , test_options ]
```

**Options**

See Options for test in [R] test.

**Remarks and examples**

See [SEM] Example 8 and [SEM] Example 16.

test works in the metric of SEM, which is to say path coefficients, variances, and covariances. If you want to frame your tests in terms of standardized coefficients and correlations and you fit your model with sem, not gsem, then prefix test with estat stdize:; see [SEM] estat stdize.
Stored results

See *Stored results* in [R] test.

Also see

[SEM] Example 8 — Testing that coefficients are equal, and constraining them

[SEM] Example 16 — Correlation

[R] test — Test linear hypotheses after estimation

[SEM] estat stdize — Test standardized parameters

[SEM] estat eqtest — Equation-level tests that all coefficients are zero

[SEM] lincom — Linear combinations of parameters

[SEM] lrtest — Likelihood-ratio test of linear hypothesis
**testnl — Wald test of nonlinear hypotheses**

### Description

`testnl` is a postestimation command for use after `sem`, `gsem`, and other Stata estimation commands.

`testnl` performs the Wald test of the nonlinear hypothesis or hypotheses. In the case of `sem` and `gsem`, you must use the `_b[ ]` coefficient notation.

### Menu

Statistics > SEM (structural equation modeling) > Testing and CIs > Wald tests of nonlinear hypotheses

### Syntax

```
testnl exp = exp [ = ... ] [ , options ]
```

```
testnl (exp = exp [ = ... ]) [ (exp = exp [ = ... ]) ... ] [ , options ]
```

### Options

See `Options in [R] testnl`.

### Remarks and examples

`testnl` works in the metric of SEM, which is to say path coefficients, variances, and covariances. If you want to frame your tests in terms of standardized coefficients and correlations and you fit the model with `sem`, not `gsem`, then prefix `testnl` with `estat stdize`; see [SEM] estat stdize.

#### Technical note

`estat stdize`: is unnecessary because, with `testnl`, everywhere you wanted a standardized coefficient or correlation, you could just type the formula. If you did that, you would get the same answer except for numerical precision. In this case, the answer produced with the `estat stdize`: prefix will be a little more accurate because `estat stdize`: is able to substitute an analytic derivative in one part of the calculation where `testnl`, doing the whole thing itself, would be forced to use a numeric derivative.
Stored results

See Stored results in [R] testnl.

Also see

[R] testnl — Test nonlinear hypotheses after estimation

[SEM] test — Wald test of linear hypotheses

[SEM] lrtest — Likelihood-ratio test of linear hypothesis

[SEM] estat stdize — Test standardized parameters

[SEM] estat eqtest — Equation-level tests that all coefficients are zero

[SEM] nlcom — Nonlinear combinations of parameters
ADF, method(adf). ADF stands for asymptotic distribution free and is a method used to obtain fitted parameters for standard linear SEMs. ADF is used by `sem` when option `method(adf)` is specified. Other available methods are ML, QML, and MLMV.

anchoring, anchor variable. A variable is said to be the anchor of a latent variable if the path coefficient between the latent variable and the anchor variable is constrained to be 1. `sem` and `gsem` use anchoring as a way of normalizing latent variables and thus identifying the model.

baseline model. A baseline model is a covariance model—a model of fitted means and covariances of observed variables without any other paths—with most of the covariances constrained to be 0. That is, a baseline model is a model of fitted means and variances but typically not all the covariances. Also see saturated model. Baseline models apply only to standard linear SEMs.

Bentler–Weeks matrices. The Bentler and Weeks (1980) formulation of standard linear SEMs places the results in a series of matrices organized around how results are calculated. See `SEM estat framework`.

bootstrap, vce(bootstrap). The bootstrap is a replication method for obtaining variance estimates. Consider an estimation method $E$ for estimating $\theta$. Let $\hat{\theta}$ be the result of applying $E$ to dataset $D$ containing $N$ observations. The bootstrap is a way of obtaining variance estimates for $\hat{\theta}$ from repeated estimates $\hat{\theta}_1, \hat{\theta}_2, \ldots$, where each $\hat{\theta}_i$ is the result of applying $E$ to a dataset of size $N$ drawn with replacement from $D$. See `SEM sem option method()` and `[R] bootstrap`.

vce(bootstrap) is allowed with `sem` but not `gsem`. You can obtain bootstrap results by prefixing the `gsem` command with `bootstrap:`, but remember to specify bootstrap's `cluster()` and `idcluster()` options if you are fitting a multilevel model. See `SEM Intro 9`.

Builder. The Builder is Stata’s graphical interface for building `sem` and `gsem` models. The Builder is also known as the SEM Builder. See `SEM Intro 2`, `SEM Builder`, and `SEM Builder, generalized`.

categorical latent variable. A categorical latent variable has levels that represent unobserved groups in the population. Latent classes are identified with the levels of the categorical latent variables and may represent healthy and unhealthy individuals, consumers with different buying preferences, or different motivations for delinquent behavior. Categorical latent variables are allowed in `gsem` but not in `sem`. See `SEM gsem lclass options` and `SEM Intro 2`.

CFA, CFA models. CFA stands for confirmatory factor analysis. It is a way of analyzing measurement models. CFA models is a synonym for measurement models.

cluster, vce(cluster clustvar). Cluster is the name we use for the generalized Huber/White/sandwich estimator of the VCE, which is the robust technique generalized to relax the assumption that errors are independent across observations to be that they are independent across clusters of observations. Within cluster, errors may be correlated.

Clustered standard errors are reported when `sem` or `gsem` option `vce(cluster clustvar)` is specified. The other available techniques are OIM, OPG, robust, bootstrap, and jackknife. Also available for `sem` only is EIM.

coefficient of determination. The coefficient of determination is the fraction (or percentage) of variation (variance) explained by an equation of a model. The coefficient of determination is thus like $R^2$ in linear regression.
command language. Stata’s `sem` and `gsem` commands provide a way to specify SEMs. The alternative is to use the Builder to draw path diagrams; see `[SEM] Intro 2`, `[SEM] Builder`, and `[SEM] Builder, generalized.

complementary log-log regression. Complementary log-log regression is a term for generalized linear response functions that are family Bernoulli, link clog-log. It is used for binary outcome data. Complementary log-log regression is also known in Stata circles as clog-log regression or just clog-log. See generalized linear response functions.

conditional normality assumption. See normality assumption, joint and conditional.

constraints. See parameter constraints.

continuous latent variable. A continuous latent variable is an unobserved variable, such as mathematical ability, with values that are assumed to follow a continuous distribution. Both `sem` and `gsem` allow continuous latent variables that are assumed to follow a Gaussian distribution with a mean and variance that are either estimated or constrained to a specific value for identification. See identification.

correlated uniqueness model. A correlated uniqueness model is a kind of measurement model in which the errors of the measurements have a structured correlation. See `[SEM] Intro 5`.

crossed-effects models. See multilevel models.

curved path. See path.

degree-of-freedom adjustment. In estimates of variances and covariances, a finite-population degree-of-freedom adjustment is sometimes applied to make the estimates unbiased.

Let’s write an estimated variance as $\hat{\sigma}_{ii}$ and write the “standard” formula for the variance as $\hat{\sigma}_{ii} = S_{ii}/N$. If $\hat{\sigma}_{ii}$ is the variance of observable variable $x_i$, it can readily be proven that $S_{ii}/N$ is a biased estimate of the variances in samples of size $N$ and that $S_{ii}/(N - 1)$ is an unbiased estimate. It is usual to calculate variances with $S_{ii}/(N - 1)$, which is to say the “standard” formula has a multiplicative degree-of-freedom adjustment of $N/(N - 1)$ applied to it.

If $\hat{\sigma}_{ii}$ is the variance of estimated parameter $\beta_i$, a similar finite-population degree-of-freedom adjustment can sometimes be derived that will make the estimate unbiased. For instance, if $\beta_i$ is a coefficient from a linear regression, an unbiased estimate of the variance of regression coefficient $\beta_i$ is $S_{ii}/(N - p - 1)$, where $p$ is the total number of regression coefficients estimated excluding the intercept. In other cases, no such adjustment can be derived. Such estimators have no derivable finite-sample properties, and one is left only with the assurances provided by its provable asymptotic properties. In such cases, the variance of coefficient $\beta_i$ is calculated as $S_{ii}/N$, which can be derived on theoretical grounds. SEM is an example of such an estimator.

SEM is a remarkably flexible estimator and can reproduce results that can sometimes be obtained by other estimators. SEM might produce asymptotically equivalent results, or it might produce identical results depending on the estimator. Linear regression is an example in which `sem` and `gsem` produce the same results as `regress`. The reported standard errors, however, will not look identical because the linear-regression estimates have the finite-population degree-of-freedom adjustment applied to them and the SEM estimates do not. To see the equivalence, you must undo the adjustment on the reported linear regression standard errors by multiplying them by $\sqrt{((N - p - 1)/N)}$.

direct, indirect, and total effects. Consider the following system of equations:

$$
\begin{align*}
y_1 &= b_{10} + b_{11}y_2 + b_{12}x_1 + b_{13}x_3 + e_1 \\
y_2 &= b_{20} + b_{21}y_3 + b_{22}x_1 + b_{23}x_4 + e_2 \\
y_3 &= b_{30} + b_{32}x_1 + b_{33}x_5 + e_3
\end{align*}
$$
The total effect of $x_1$ on $y_1$ is $b_{12} + b_{11}b_{22} + b_{11}b_{21}b_{32}$. It measures the full change in $y_1$ based on allowing $x_1$ to vary throughout the system.

The direct effect of $x_1$ on $y_1$ is $b_{12}$. It measures the change in $y_1$ caused by a change in $x_1$ holding other endogenous variables—namely, $y_2$ and $y_3$—constant.

The indirect effect of $x_1$ on $y_1$ is obtained by subtracting the total and direct effect and is thus $b_{11}b_{22} + b_{11}b_{21}b_{32}$.

**EIM, vce(eim)**. EIM stands for expected information matrix, defined as the inverse of the negative of the expected value of the matrix of second derivatives, usually of the log-likelihood function. The EIM is an estimate of the VCE. EIM standard errors are reported when `sem` option `vce(eim)` is specified. EIM is available only with `sem`. The other available techniques for `sem` are OIM, OPG, robust, cluster, bootstrap, and jackknife.

**endogenous variable**. A variable, observed or latent, is endogenous (determined by the system) if any path points to it. Also see **exogenous variable**.

**error, error variable**. The error is random disturbance $e$ in a linear equation:

$$y = b_0 + b_1x_1 + b_2x_2 + \cdots + e$$

An error variable is an unobserved exogenous variable in path diagrams corresponding to $e$. Mathematically, error variables are just another example of latent exogenous variables, but in `sem` and `gsem`, error variables are considered to be in a class by themselves. All (Gaussian) endogenous variables—observed and latent—have a corresponding error variable. Error variables automatically and inalterably have their path coefficients fixed to be 1. Error variables have a fixed naming convention in the software. If a variable is the error for (observed or latent) endogenous variable $y$, then the residual variable’s name is `e.y`.

In `sem` and `gsem`, error variables are uncorrelated with each other unless explicitly indicated otherwise. That indication is made in path diagrams by drawing a curved path between the error variables and is indicated in command notation by including `cov(e.name1*e.name2)` among the options specified on the `sem` command. In `gsem`, errors for family Gaussian, link log responses are not allowed to be correlated.

**estimation method**. There are a variety of ways that one can solve for the parameters of an SEM. Different methods make different assumptions about the data-generation process, so it is important that you choose a method appropriate for your model and data; see [SEM] Intro 4.

**exogenous variable**. A variable, observed or latent, is exogenous (determined outside the system) if paths only originate from it or, equivalently, no path points to it. In this manual, we do not distinguish whether exogenous variables are strictly exogenous—that is, uncorrelated with the errors. Also see **endogenous variable**.

**family distribution**. See **generalized linear response functions**.

**fictional data**. Fictional data are data that have no basis in reality even though they might look real; they are data that are made up for use in examples.

**finite mixture model**. A finite mixture model (FMM) is a latent class model in which parameters of a regression model are allowed to vary across classes. The regression model may have a linear or generalized linear response function.

**first- and second-order latent variables**. If a latent variable is measured by other latent variables only, the latent variable that does the measuring is called first-order latent variable, and the latent variable being measured is called the second-order latent variable.
first-, second-, and higher-level (latent) variables. Consider a multilevel model of patients within doctors within hospitals. First-level variables are variables that vary at the observational (patient) level. Second-level variables vary across doctors but are constant within doctors. Third-level variables vary across hospitals but are constant within hospitals. This jargon is used whether variables are latent or not.

full joint and conditional normality assumption. See normality assumption, joint and conditional.

gamma regression. Gamma regression is a term for generalized linear response functions that are family gamma, link log. It is used for continuous, nonnegative, positively skewed data. Gamma regression is also known as log-gamma regression. See generalized linear response functions.

Gaussian regression. Gaussian regression is another term for linear regression. It is most often used when referring to generalized linear response functions. In that framework, Gaussian regression is family Gaussian, link identity. See generalized linear response functions.

generalized linear response functions. Generalized linear response functions include linear functions and include functions such as probit, logit, multinomial logit, ordered probit, ordered logit, Poisson, and more.

These generalized linear functions are described by a link function $g(\cdot)$ and statistical distribution $F$. The link function $g(\cdot)$ specifies how the response variable $y_i$ is related to a linear equation of the explanatory variables, $x_i\beta$, and the family $F$ specifies the distribution of $y_i$:

$$g\{E(y_i)\} = x_i\beta, \quad y_i \sim F$$

If we specify that $g(\cdot)$ is the identity function and $F$ is the Gaussian (normal) distribution, then we have linear regression. If we specify that $g(\cdot)$ is the logit function and $F$ the Bernoulli distribution, then we have logit (logistic) regression.

In this generalized linear structure, the family may be Gaussian, gamma, Bernoulli, binomial, Poisson, negative binomial, ordinal, or multinomial. The link function may be the identity, log, logit, probit, or complementary log-log.

$gsem$ fits models with generalized linear response functions.

generalized method of moments. Generalized method of moments (GMM) is a method used to obtain fitted parameters. In this documentation, GMM is referred to as ADF, which stands for asymptotic distribution free and is available for use with $sem$. Other available methods for use with $sem$ are ML, QML, ADF, and MLMV.

The SEM moment conditions are cast in terms of second moments, not the first moments used in many other applications associated with GMM.

generalized SEM. Generalized SEM is a term we have coined to mean SEM optionally allowing generalized linear response functions, multilevel models, or categorical latent variables. $gsem$ fits generalized SEMs.

GMM. See generalized method of moments.

goodness-of-fit statistic. A goodness-of-fit statistic is a value designed to measure how well the model reproduces some aspect of the data the model is intended to fit. SEM reproduces the first- and second-order moments of the data, with an emphasis on the second-order moments, and thus goodness-of-fit statistics appropriate for use after $sem$ compare the predicted covariance matrix (and mean vector) with the matrix (and vector) observed in the data.

$gsem$. $gsem$ is the Stata command that fits generalized SEMs. Also see $sem$.

GUI. See Builder.
identification. Identification refers to the conceptual constraints on parameters of a model that are required for the model’s remaining parameters to have a unique solution. A model is said to be unidentified if these constraints are not supplied. These constraints are of two types: substantive constraints and normalization constraints.

Normalization constraints deal with the problem that one scale works as well as another for each continuous latent variable in the model. One can think, for instance, of propensity to write software as being measured on a scale of 0 to 1, 1 to 100, or any other scale. The normalization constraints are the constraints necessary to choose one particular scale. The normalization constraints are provided automatically by `sem` and `gsem` by anchoring with unit loadings.

Substantive constraints are the constraints you specify about your model so that it has substantive content. Usually, these constraints are 0 constraints implied by the paths omitted, but they can include explicit parameter constraints as well. It is easy to write a model that is not identified for substantive reasons; see [SEM] Intro 4.

indicator variables, indicators. The term “indicator variable” has two meanings. An indicator variable is a 0/1 variable that contains whether something is true. The other usage is as a synonym for measurement variables.

indirect effects. See direct, indirect, and total effects.

initial values. See starting values.

intercept. An intercept for the equation of endogenous variable $y$, observed or latent, is the path coefficient from `_cons` to $y$. `_cons` is Stata-speak for the built-in variable containing 1 in all observations. In SEM-speak, `_cons` is an observed exogenous variable.

jackknife, vce(jackknife). The jackknife is a replication method for obtaining variance estimates. Consider an estimation method $E$ for estimating $\theta$. Let $\hat{\theta}$ be the result of applying $E$ to dataset $D$ containing $N$ observations. The jackknife is a way of obtaining variance estimates for $\hat{\theta}$ from repeated estimates $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_N$, where each $\hat{\theta}_i$ is the result of applying $E$ to $D$ with observation $i$ removed. See [SEM] sem option method() and [R] jackknife.

vce(jackknife) is allowed with `sem` but not `gsem`. You can obtain jackknife results by prefixing the `gsem` command with `jackknife:`, but remember to specify `jackknife`’s `cluster()` and `idcluster()` options if you are fitting a multilevel model. See [SEM] Intro 9.

joint normality assumption. See normality assumption, joint and conditional.

Lagrange multiplier test. Synonym for score test.

latent class analysis. Latent class analysis is useful for identifying and understanding unobserved groups in a population. When performing a latent class analysis, we fit models that include a categorical latent variable with levels called latent classes that correspond to the unobserved groups. In latent class analysis, we can compare models with differing numbers of latent classes and different sets of constraints on parameters to determine the best-fitting model. For a given model, we can compare parameter estimates across classes. We can estimate the proportion of the population in each latent class. And we can predict the probabilities that the individuals in our sample belong to each latent class.

latent class model. Any model with a categorical latent variable that is fit as part of a latent class analysis. In some literature, latent class models are more narrowly defined to include only categorical latent variables and the binary or categorical observed variables that are indicators of class membership, but we do not make such a restriction. See [SEM] Intro 5.

latent cluster model. A type of latent class model with continuous observed outcomes.
latent growth model. A latent growth model is a kind of measurement model in which the observed values are collected over time and are allowed to follow a trend. See [SEM] Intro 5.

latent profile model. A type of latent class model with continuous observed outcomes.

latent variable. A variable is latent if it is not observed. A variable is latent if it is not in your dataset but you wish it were. You wish you had a variable recording the propensity to commit violent crime, or socioeconomic status, or happiness, or true ability, or even income accurately recorded. Latent variables are sometimes described as imagined variables.

In the software, latent variables are usually indicated by having at least their first letter capitalized.

Also see continuous latent variable, categorical latent variable, first- and second-order latent variables, first-, second-, and higher-level (latent) variables, and observed variables.

linear regression. Linear regression is a kind of SEM in which there is a single equation. See [SEM] Intro 5.

link function. See generalized linear response functions.

logit regression. Logit regression is a term for generalized linear response functions that are family Bernoulli, link logit. It is used for binary outcome data. Logit regression is also known as logistic regression and also simply as logit. See generalized linear response functions.

manifest variables. Synonym for observed variables.

measure, measurement, x a measurement of X, x measures X. See measurement variables.

measurement models, measurement component. A measurement model is a particular kind of model that deals with the problem of translating observed values to values suitable for modeling. Measurement models are often combined with structural models and then the measurement model part is referred to as the measurement component. See [SEM] Intro 5.

measurement variables, measure, measurement, x a measurement of X, x measures X. Observed variable x is a measurement of latent variable X if there is a path connecting x ← X. Measurement variables are modeled by measurement models. Measurement variables are also called indicator variables.

method. Method is just an English word and should be read in context. Nonetheless, method is used here usually to refer to the method used to solve for the fitted parameters of an SEM. Those methods are ML, QML, MLMV, and ADF. Also see technique.

MIMIC. See multiple indicators and multiple causes.

mixed-effects models. See multilevel models.

ML, method(ml). ML stands for maximum likelihood. It is a method to obtain fitted parameters. ML is the default method used by sem and gsem. Other available methods for sem are QML, MLMV, and ADF. Also available for gsem is QML.

MLMV, method(mlmv). MLMV stands for maximum likelihood with missing values. It is an ML method used to obtain fitted parameters in the presence of missing values. MLMV is the method used by sem when the method(mlmv) option is specified; method(mlmv) is not available with gsem. Other available methods for use with sem are ML, QML, and ADF. These other methods omit from the calculation observations that contain missing values.

modification indices. Modification indices are score tests for adding paths where none appear. The paths can be for either coefficients or covariances.

moments (of a distribution). The moments of a distribution are the expected values of powers of a random variable or centralized (demeaned) powers of a random variable. The first moments are
the expected or observed means, and the second moments are the expected or observed variances and covariances.

**multilevel models.** Multilevel models are models that include unobserved effects (latent variables) for different groups in the data. For instance, in a dataset of students, groups of students might share the same teacher. If the teacher’s identity is recorded in the data, then one can introduce a latent variable that is constant within teacher and that varies across teachers. This is called a two-level model.

If teachers could in turn be grouped into schools, and school identities were recorded in the data, then one can introduce another latent variable that is constant within school and varies across schools. This is called a three-level (nested-effects) model.

In the above example, observations (students) are said to be nested within teacher nested within school. Sometimes there is no such subsequent nesting structure. Consider workers nested within occupation and industry. The same occupations appear in various industries and the same industries appear within various occupations. We can still introduce latent variables at the occupation and industry level. In such cases, the model is called a crossed-effects model.

The latent variables that we have discussed are also known as random effects. Any coefficients on observed variables in the model are known as the fixed portion of the model. Models that contain fixed and random portions are known as mixed-effects models.

**multinomial logit regression.** Multinomial logit regression is a term for generalized linear response functions that are family multinomial, link logit. It is used for categorical-outcome data when the outcomes cannot be ordered. Multinomial logit regression is also known as multinomial logistic regression and as mlogit in Stata circles. See [*generalized linear response functions*](#).

**multiple correlation.** The multiple correlation is the correlation between endogenous variable $y$ and its linear prediction.

**multiple indicators and multiple causes.** Multiple indicators and multiple causes is a kind of structural model in which observed causes determine a latent variable, which in turn determines multiple indicators. See [*SEM* Intro 5](#).

**multivariate regression.** Multivariate regression is a kind of structural model in which each member of a set of observed endogenous variables is a function of the same set of observed exogenous variables and a unique random disturbance term. The disturbances are correlated. Multivariate regression is a special case of *seemingly unrelated regression*.

**negative binomial regression.** Negative binomial regression is a term for generalized linear response functions that are family negative binomial, link log. It is used for count data that are overdispersed relative to Poisson. Negative binomial regression is also known as nbreg in Stata circles. See [*generalized linear response functions*](#).

**nested-effects models.** See [*multilevel models*](#).

**nonrecursive (structural) model (system), recursive (structural) model (system).** A structural model (system) is said to be nonrecursive if there are paths in both directions between one or more pairs of endogenous variables. A system is recursive if it is a system—it has endogenous variables that appear with paths from them—and it is not nonrecursive.

A nonrecursive model may be unstable. Consider, for instance,

$$y_1 = 2y_2 + 1x_1 + e_1$$

$$y_2 = 3y_1 - 2x_2 + e_2$$
This model is unstable. To see this, without loss of generality, treat \( x_1 + e_1 \) and \( 2x_2 + e_2 \) as if they were both 0. Consider \( y_1 = 1 \) and \( y_2 = 1 \). Those values result in new values \( y_1 = 2 \) and \( y_2 = 3 \), and those result in new values \( y_1 = 6 \) and \( y_2 = 6 \), and those result in new values . . . .

Continue in this manner, and you reach infinity for both endogenous variables. In the jargon of the mathematics used to check for this property, the eigenvalues of the coefficient matrix lie outside the unit circle.

On the other hand, consider these values:

\[
\begin{align*}
y_1 &= 0.5y_2 + 1x_1 + e_1 \\
y_2 &= 1.0y_1 - 2x_2 + e_2
\end{align*}
\]

These results are stable in that the resulting values converge to \( y_1 = 0 \) and \( y_2 = 0 \). In the jargon of the mathematics used to check for this property, the eigenvalues of the coefficient matrix lie inside the unit circle.

Finally, consider the values

\[
\begin{align*}
y_1 &= 0.5y_2 + 1x_1 + e_1 \\
y_2 &= 2.0y_1 - 2x_2 + e_2
\end{align*}
\]

Start with \( y_1 = 1 \) and \( y_2 = 1 \). That yields new values \( y_1 = 0.5 \), and \( y_2 = 2 \) and that yields new values \( y_1 = 1 \) and \( y_2 = 1 \), and that yields new values \( y_1 = 0.5 \) and \( y_2 = 2 \), and it will oscillate forever. In the jargon of the mathematics used to check for this property, the eigenvalues of the coefficient matrix lie on the unit circle. These coefficients are also considered to be unstable.

**normality assumption, joint and conditional.** The derivation of the standard, linear SEM estimator usually assumes the full joint normality of the observed and latent variables. However, full joint normality can replace the assumption of normality conditional on the values of the exogenous variables, and all that is lost is one goodness-of-fit test (the test reported by `sem` on the output) and the justification for use of optional method MLMV for dealing with missing values. This substitution of assumptions is important for researchers who cannot reasonably assume normality of the observed variables. This includes any researcher including, say, variables age and age-squared in his or her model.

Meanwhile, the generalized SEM makes only the conditional normality assumption.

Be aware that even though the full joint normality assumption is not required for the standard linear SEM, `sem` calculates the log-likelihood value under that assumption. This is irrelevant except that log-likelihood values reported by `sem` cannot be compared with log-likelihood values reported by `gsem`, which makes the lesser assumption.

See [SEM] Intro 4.

**normalization constraints.** See identification.

**normalized residuals.** See standardized residuals.

**observed variables.** A variable is observed if it is a variable in your dataset. In this documentation, we often refer to observed variables by using \( x_1, x_2, \ldots, y_1, y_2 \), and so on; in reality, observed variables have names such as \( mpg, weight, testscore \), and so on.

In the software, observed variables are usually indicated by having names that are all lowercase. Also see latent variable.
OIM, vce(oim). OIM stands for observed information matrix, defined as the inverse of the negative of the matrix of second derivatives, usually of the log likelihood function. The OIM is an estimate of the VCE. OIM is the default VCE that \texttt{sem} and \texttt{gsem} report. The other available techniques are EIM, OPG, robust, cluster, bootstrap, and jackknife.

OPG, vce(opg). OPG stands for outer product of the gradients, defined as the cross product of the observation-level first derivatives, usually of the log likelihood function. The OPG is an estimate of the VCE. The other available techniques are OIM, EIM, robust, cluster, bootstrap, and jackknife.

ordered complementary log-log regression. Ordered complementary log-log regression is a term for generalized linear response functions that are family ordinal, link clog-log. It is used for ordinal-outcome data. Ordered complementary log-log regression is also known as ocln-log in Stata circles. See \textit{generalized linear response functions}.

ordered logit regression. Ordered logit regression is a term for generalized linear response functions that are family ordinal, link logit. It is used for ordinal outcome data. Ordered logit regression is also known as ordered logistic regression, as just ordered logit, and as ologit in Stata circles. See \textit{generalized linear response functions}.

ordered probit regression. Ordered probit regression is a term for generalized linear response functions that are family ordinal, link probit. It is used for ordinal-outcome data. Ordered probit regression is also known as just ordered probit and known as oprobit in Stata circles. See \textit{generalized linear response functions}.

parameter constraints. Parameter constraints are restrictions placed on the parameters of the model. These constraints are typically in the form of 0 constraints and equality constraints. A 0 constraint is implied, for instance, when no path is drawn connecting \(x\) with \(y\). An equality constraint is specified when one path coefficient is forced to be equal to another or one covariance is forced to be equal to another.

Also see \textit{identification}.

parameters, ancillary parameters. The parameters are the to-be-estimated coefficients of a model. These include all path coefficients, means, variances, and covariances. In mathematical notation, the theoretical parameters are often written as \(\theta = (\alpha, \beta, \mu, \Sigma)\), where \(\alpha\) is the vector of intercepts, \(\beta\) is the vector of path coefficients, \(\mu\) is the vector of means, and \(\Sigma\) is the matrix of variances and covariances. The resulting parameter estimates are written as \(\hat{\theta}\).

Ancillary parameters are extra parameters beyond the ones just described that concern the distribution. These include the scale parameter of gamma regression, the dispersion parameter for negative binomial regression, and the cutpoints for ordered probit, logit, and clog-log regression, and the like. These parameters are also included in \(\theta\).

path. A path, typically shown as an arrow drawn from one variable to another, states that the first variable determines the second variable, at least partially. If \(x \rightarrow y\), or equivalently \(y \leftarrow x\), then \(y_j = \alpha + \cdots + \beta x_j + \cdots + e.y_j\), where \(\beta\) is said to be the \(x \rightarrow y\) path coefficient. The ellipses are included to account for paths to \(y\) from other variables. \(\alpha\) is said to be the intercept and is automatically added when the first path to \(y\) is specified.

A curved path is a curved line connecting two variables, and it specifies that the two variables are allowed to be correlated. If there is no curved path between variables, the variables are usually assumed to be uncorrelated. We say usually because correlation is assumed among observed exogenous variables and, in the command language, assumed among latent exogenous variables, and if some of the correlations are not desired, they must be suppressed. Many authors refer to covariances rather than correlations. Strictly speaking, the curved path denotes a nonzero covariance. A correlation is often called a \textit{standardized covariance}. 

A curved path can connect a variable to itself, and in that case, it indicates a variance. In path diagrams in this manual, we typically do not show such variance paths even though variances are assumed.

**path coefficient.** The path coefficient is associated with a path; see path. Also see intercept.

**path diagram.** A path diagram is a graphical representation that shows the relationships among a set of variables using paths. See [SEM Intro 2] for a description of path diagrams.

**path notation.** Path notation is a syntax defined by the authors of Stata’s `sem` and `gsem` commands for entering path diagrams in a command language. Models to be fit may be specified in path notation or they may be drawn using path diagrams into the Builder.

**Poisson regression.** Poisson regression is a term for generalized linear response functions that are family Poisson, link log. It is used for count data. See generalized linear response functions.

**probit regression.** Probit regression is a term for generalized linear response functions that are family Bernoulli, link probit. It is used for binary outcome data. Probit regression is also known simply as probit. See generalized linear response functions.

**p-value.** P-value is another term for the reported significance level associated with a test. Small p-values indicate rejection of the null hypothesis.

**QML, method(ml) vce(robust).** QML stands for quasimaximum likelihood. It is a method used to obtain fitted parameters and a technique used to obtain the corresponding VCE. QML is used by `sem` and `gsem` when options `method(ml)` and `vce(robust)` are specified. Other available methods are ML, MLMV, and ADF. Other available techniques are OIM, EIM, OPG, cluster, bootstrap, and jackknife.

**quadrature.** Quadrature is generic method for performing numerical integration. `gsem` uses quadrature in any model including latent variables (excluding error variables). `sem`, being limited to linear models, does not need to perform quadrature.

**random-effects models.** See [multilevel models].

**regression.** A regression is a model in which an endogenous variable is written as a function of other variables, parameters to be estimated, and a random disturbance.

**reliability.** Reliability is the proportion of the variance of a variable not due to measurement error. A variable without measure error has reliability 1.

**residual.** In this manual, we reserve the word “residual” for the difference between the observed and fitted moments of an SEM. We use the word “error” for the disturbance associated with a (Gaussian) linear equation; see error. Also see standardized residuals.

**robust, vce(robust).** Robust is the name we use here for the Huber/White/sandwich estimator of the VCE. This technique requires fewer assumptions than most other techniques. In particular, it merely assumes that the errors are independently distributed across observations and thus allows the errors to be heteroskedastic. Robust standard errors are reported when the `sem` (`gsem`) option `vce(robust)` is specified. The other available techniques are OIM, EIM, OPG, cluster, bootstrap, and jackknife.

**saturated model.** A saturated model is a full covariance model—a model of fitted means and covariances of observed variables without any restrictions on the values. Also see baseline model. Saturated models apply only to standard linear SEMs.

**score test, Lagrange multiplier test.** A score test is a test based on first derivatives of a likelihood function. Score tests are especially convenient for testing whether constraints on parameters should be relaxed or parameters should be added to a model. Also see Wald test.
scores. Scores has two unrelated meanings. First, scores are the observation-by-observation first-derivatives of the (quasi) log-likelihood function. When we use the word “scores”, this is what we mean. Second, in the factor-analysis literature, scores (usually in the context of factor scores) refers to the expected value of a latent variable conditional on all the observed variables. We refer to this simply as the predicted value of the latent variable.

second-level latent variable. See first-, second-, and higher-order latent variables.

second-order latent variable. See first- and second-order latent variables.

seemingly unrelated regression. Seemingly unrelated regression is a kind of structural model in which each member of a set of observed endogenous variables is a function of a set of observed exogenous variables and a unique random disturbance term. The disturbances are correlated and the sets of exogenous variables may overlap. If the sets of exogenous variables are identical, this is referred to as multivariate regression.

SEM. SEM stands for structural equation modeling and for structural equation model. We use SEM in capital letters when writing about theoretical or conceptual issues as opposed to issues of the particular implementation of SEM in Stata with the sem or gsem commands.

sem. sem is the Stata command that fits standard linear SEMs. Also see gsem.

SSD, ssd. See summary statistics data.

standard linear SEM. An SEM without multilevel effects in which all response variables are given by a linear equation. Standard linear SEM is what most people mean when they refer to just SEM. Standard linear SEMs are fit by sem, although they can also be fit by gsem; see generalized SEM.

standardized coefficient. In a linear equation $y = \ldots bx + \ldots$, the standardized coefficient $\beta$ is $(\hat{\sigma}_y/\hat{\sigma}_x)b$. Standardized coefficients are scaled to units of standard deviation change in $y$ for a standard deviation change in $x$.

standardized covariance. A standardized covariance between $y$ and $x$ is equal to the correlation of $y$ and $x$, that is, it is equal to $\sigma_{xy}/\sigma_x\sigma_y$. The covariance is equal to the correlation when variables are standardized to have variance 1.

standardized residuals, normalized residuals. Standardized residuals are residuals adjusted so that they follow a standard normal distribution. The difficulty is that the adjustment is not always possible. Normalized residuals are residuals adjusted according to a different formula that roughly follow a standard normal distribution. Normalized residuals can always be calculated.

starting values. The estimation methods provided by sem and gsem are iterative. The starting values are values for each of the parameters to be estimated that are used to initial the estimation process. sem and gsem provide starting values automatically, but in some cases, these are not good enough and you must both diagnose the problem and provide better starting values. See [SEM] Intro 12.

structural equation model. Different authors use the term “structural equation model” in different ways, but all would agree that an SEM sometimes carries the connotation of being a structural model with a measurement component, that is, combined with a measurement model.

structural model. A structural model is a model in which the parameters are not merely a description but are believed to be of a causal nature. Obviously, SEM can fit structural models and thus so can sem and gsem. Neither SEM, sem, nor gsem are limited to fitting structural models, however.

Structural models often have multiple equations and dependencies between endogenous variables, although that is not a requirement.

See [SEM] Intro 5. Also see structural equation model.

structured (correlation or covariance). See unstructured and structured (correlation or covariance).
**substantive constraints.** See *identification.*

**summary statistics data.** Data are sometimes available only in summary statistics form, as means and covariances; means, standard deviations or variances, and correlations; covariances; standard deviations or variances and correlations; or correlations. SEM can be used to fit models with such data in place of the underlying raw data. The *ssd* command creates datasets containing summary statistics.

**technique.** Technique is just an English word and should be read in context. Nonetheless, technique is usually used here to refer to the technique used to calculate the estimated VCE. Those techniques are OIM, EIM, OPG, robust, cluster, bootstrap, and jackknife.

Technique is also used to refer to the available techniques used with *ml*, Stata’s optimizer and likelihood maximizer, to find the solution.

**total effects.** See *direct, indirect, and total effects.*

**unstandardized coefficient.** A coefficient that is not standardized. If \( \text{mpg} = -0.006 \times \text{weight} + 39.44028 \), then \(-0.006\) is an unstandardized coefficient and, as a matter of fact, is measured in mpg-per-pound units.

**unstructured and structured (correlation or covariance).** A set of variables, typically error variables, is said to have an unstructured correlation or covariance if the covariance matrix has no particular pattern imposed by theory. If a pattern is imposed, the correlation or covariance is said to be structured.

**variance–covariance matrix of the estimator.** The estimator is the formula used to solve for the fitted parameters, sometimes called the fitted coefficients. The VCE is the estimated variance–covariance matrix of the parameters. The diagonal elements of the VCE are the variances of the parameters or equivalent; the square roots of those elements are the reported standard errors of the parameters.

VCE. See *variance–covariance matrix of the estimator.*

**Wald test.** A Wald test is a statistical test based on the estimated variance–covariance matrix of the parameters. Wald tests are especially convenient for testing possible constraints to be placed on the estimated parameters of a model. Also see *score test.*

**weighted least squares.** Weighted least squares (WLS) is a method used to obtain fitted parameters. In this documentation, WLS is referred to as ADF, which stands for asymptotic distribution free. Other available methods are ML, QML, and MLMV. ADF is, in fact, a specific kind of the more generic WLS.

WLS. See *weighted least squares.*

**Reference**

See the combined subject index and the combined author index in the *Glossary and Index*. 