Title

stata.com

gsem path notation extensions — Command syntax for path diagrams

Description Syntax Options Remarks and examples Also see

Description

This entry concerns `gsem` only.

The command syntax for describing generalized SEMs is fully specified by *paths*, `covariance()`, `variance()`, and `means()`; see [SEM] **sem and gsem path notation**.

With `gsem`, the notation is extended to allow for generalized linear response variables and to allow for multilevel latent variables. That is the subject of this entry.

Syntax

`gsem` *paths* ... [`, covariance() variance() means()`]

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()`, `variance()`, and `means()` are described in [SEM] **sem and gsem path notation**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

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 *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.

`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 teacher nested within school.

Let's assume that we correspondingly have three identification (ID) variables. We number the following list with the nesting level of the data:

3. Variable `school` contains a school ID number. If two observations have the same value of `school`, then both of those students attended the same school.

2. Variable `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, 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

Level	Latent-variable name
occupation	*Lname*`[occupation]`
industry	*Lname*`[industry]`
employee (observational)	*Lname*

Specifying family and link

`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 `family()` and `link()`, but those options are seldom used in favor of other family-and-link shorthand options such as `logit`, which means `family(bernoulli)` and `link(logit)`. These options are explained in [SEM] **gsem family-and-link options**.

In the command language, you can specify these options among the shared options at the end of a `gsem` command:

```
. gsem ..., ... logit ...
```

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

```
. gsem (y <- x1 x2), logit
```

has the same effect as typing

```
. gsem (y <- x1 x2, logit)
```

Thus you can type

```
. gsem (y1 <- x1 L, logit) (y2 <- x2 L, poisson) ..., ...
```

The y1 equation would be logit, and the y2 equation would be Poisson. If you wanted y2 to be linear regression, you could type

```
. gsem (y1 <- x1 L, logit) (y2 <- x2 L, regress) ..., ...
```

or you could be silent and let y2 default to linear regression,

```
. gsem (y1 <- x1 L, logit) (y2 <- x2 L) ..., ...
```

Also see

[SEM] **gsem** — Generalized 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