

Example 29g — Two-parameter logistic IRT model

Description

Remarks and examples

References

Also see

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/r19/gsem_cfa  
(Fictional math abilities data)
```

```
. summarize
```

Variable	Obs	Mean	Std. dev.	Min	Max
school	500	10.5	5.772056	1	20
id	500	50681.71	29081.41	71	100000
q1	500	.506	.5004647	0	1
q2	500	.394	.4891242	0	1
q3	500	.534	.4993423	0	1
q4	500	.424	.4946852	0	1
q5	500	.49	.5004006	0	1
q6	500	.434	.4961212	0	1
q7	500	.52	.5001002	0	1
q8	500	.494	.5004647	0	1
att1	500	2.946	1.607561	1	5
att2	500	2.948	1.561465	1	5
att3	500	2.84	1.640666	1	5
att4	500	2.91	1.566783	1	5
att5	500	3.086	1.581013	1	5
test1	500	75.548	5.948653	55	93
test2	500	80.556	4.976786	65	94
test3	500	75.572	6.677874	50	94
test4	500	74.078	8.845587	43	96

```
. 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

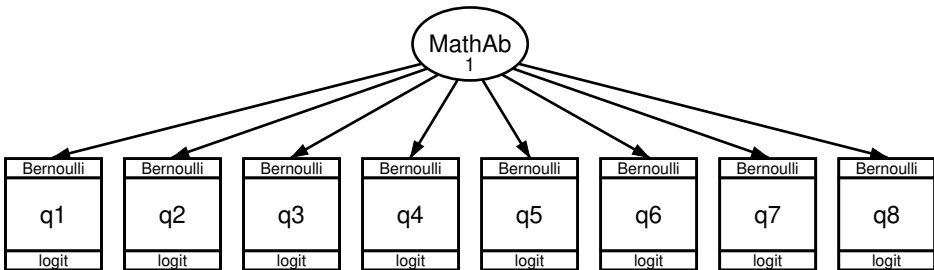
Fitting the model with the Builder

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



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
```

		Coefficient	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

q4							
MathAb	.4839501	.1310028	3.69	0.000	.2271893	.7407109	
_cons	-.3230667	.0957984	-3.37	0.001	-.5108281	-.1353054	
q5							
MathAb	1.232244	.2075044	5.94	0.000	.8255426	1.638945	
_cons	-.0494684	.1163093	-0.43	0.671	-.2774304	.1784937	
q6							
MathAb	.946535	.1707729	5.54	0.000	.6118262	1.281244	
_cons	-.3147231	.1083049	-2.91	0.004	-.5269969	-.1024493	
q7							
MathAb	1.197317	.2029485	5.90	0.000	.7995449	1.595088	
_cons	.1053405	.1152979	0.91	0.361	-.1206393	.3313203	
q8							
MathAb	.8461858	.1588325	5.33	0.000	.5348799	1.157492	
_cons	-.026705	.1034396	-0.26	0.796	-.2294429	.1760329	
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)
(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
```

	question	diff	disc	rank_d~f	rank_d~c
1.	q1	-0.0255	1.4666	3	8
2.	q2	0.8242	0.5597	8	2
3.	q3	-0.2094	0.7324	1	3
4.	q4	0.6676	0.4840	7	1
5.	q5	0.0401	1.2322	5	7
6.	q6	0.3325	0.9465	6	5
7.	q7	-0.0880	1.1973	2	6
8.	q8	0.0316	0.8462	4	4

```
. 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:_cons]`, `_b[q2:_cons]`,
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:

```
. gsem, coeflegend
Generalized structural equation model                      Number of obs = 500
(output omitted)
Log likelihood = -2637.3759
( 1)  [/]var(MathAb) = 1
```

		Coefficient	Legend
q1	MathAb	1.466636	_b[q1:MathAb]
	_cons	.0373363	_b[q1:_cons]
q2	MathAb	.5597118	_b[q2:MathAb]
	_cons	-.4613391	_b[q2:_cons]
q3	MathAb	.73241	_b[q3:MathAb]
	_cons	.1533363	_b[q3:_cons]
q4	MathAb	.4839501	_b[q4:MathAb]
	_cons	-.3230667	_b[q4:_cons]
q5	MathAb	1.232244	_b[q5:MathAb]
	_cons	-.0494684	_b[q5:_cons]
q6	MathAb	.946535	_b[q6:MathAb]
	_cons	-.3147231	_b[q6:_cons]
q7	MathAb	1.197317	_b[q7:MathAb]
	_cons	.1053405	_b[q7:_cons]
q8	MathAb	.8461858	_b[q8:MathAb]
	_cons	-.026705	_b[q8:_cons]
var(MathAb)		1	_b[/var(MathAb)]

Graphing item characteristic curves

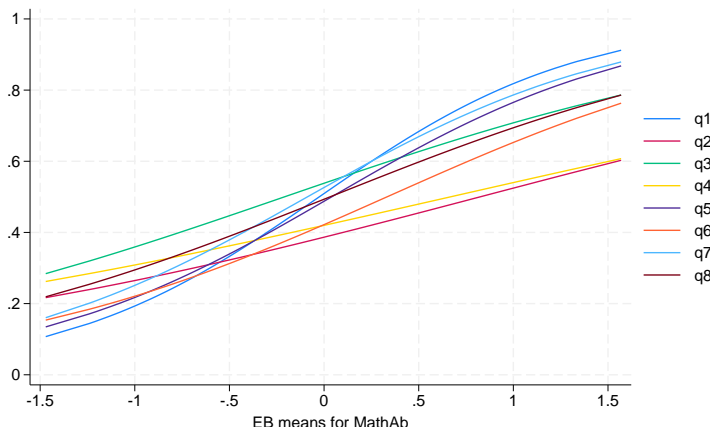
We showed you the item characteristic curves in [SEM] Example 28g, so we will show them to you again. Graphs of item characteristic curves plot the probability of a correct answer against the latent trait, which in this case is math ability.

We obtain the probabilities of a correct answer (the values of the latent variable) just as we did previously,

```
. predict pr2pl*, pr
(option conditional(ebmeans) assumed)
(using 7 quadrature points)
. predict ability2pl, latent(MathAb)
(option ebmeans assumed)
(using 7 quadrature points)
```

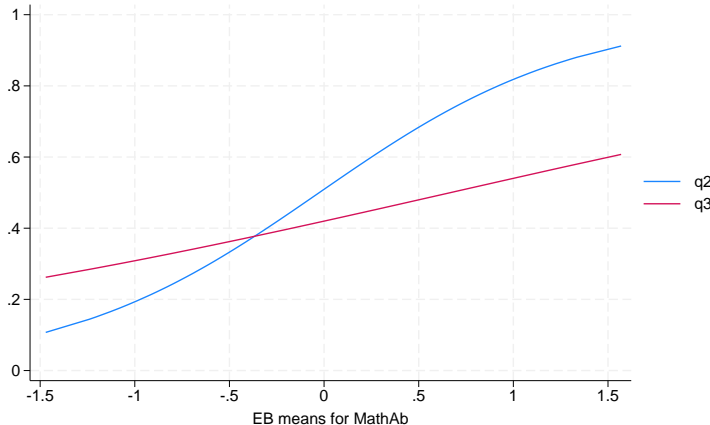
and we graph the curves just as we did previously, too. Here are all eight curves on one graph:

```
. twoway line pr2pl* ability2pl, sort xlabel(-1.5(.5)1.5)
> legend(label(1 q1) label(2 q2) label(3 q3) label(4 q4) label(5 q5)
> label(6 q6) label(7 q7) label(8 q8)) xtitle(EB means for MathAb)
```



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(label(1 q2) label(2 q3)) xtitle(EB means for MathAb)
```



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/r19/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  button.
4. Create the measurement component for MathAb.


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 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 the variance of MathAb to 1.
 - a. Choose the Select tool, .
 - b. Click on the oval for MathAb. In the Contextual Toolbar, type 1 in the σ^2 box and press *Enter*.
6. 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_irt3
```

References

- Embretson, S. E., and S. P. Reise. 2000. *Item Response Theory for Psychologists*. Mahwah, NJ: Lawrence Erlbaum.
- Rabe-Hesketh, S., A. Skrondal, and A. Pickles. 2004. Generalized multilevel structural equation modeling. *Psychometrika* 69: 167–190. <https://doi.org/10.1007/BF02295939>.
- Skrondal, A., and S. Rabe-Hesketh. 2004. *Generalized Latent Variable Modeling: Multilevel, Longitudinal, and Structural Equation Models*. Boca Raton, FL: Chapman and Hall/CRC.
- van der Linden, W. J., and R. K. Hambleton, eds. 1997. *Handbook of Modern Item Response Theory*. New York: Springer.

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

Stata, Stata Press, Mata, NetCourse, and NetCourseNow are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow is a trademark of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.



For suggested citations, see the FAQ on [citing Stata documentation](#).