Title

stata.com

irt 2pl — Two-parameter logistic model

Description

`irt 2pl` fits two-parameter logistic (2PL) models to binary items. In the 2PL model, items vary in their difficulty and discrimination.

Quick start

2PL model for binary items `b1` to `b10`

```
irt 2pl b1-b10
```

Group estimates by parameter type and sort items by difficulty

```
estat report, byparm sort(b)
```

Plot ICCs for all items

```
irtgraph icc
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irt 2pl varlist [if] [in] [weight] [, options]
```

options	Description
`group(`*varname*`)`	fit model for different groups
Model	
`cns(`*spec*`)`	apply specified parameter constraints
`listwise`	drop observations with any missing items
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim`, `robust`, `cluster` *clustvar*, `bootstrap`, or `jackknife`
Reporting	
`level(#)`	set confidence level; default is `level(95)`
`notable`	suppress coefficient table
`noheader`	suppress output header
display_options	control columns and column formats
Integration	
`intmethod(`*intmethod*`)`	integration method
`intpoints(#)`	set the number of integration points; default is `intpoints(7)`
Maximization	
maximize_options	control the maximization process; seldom used
`startvalues(`*svmethod*`)`	method for obtaining starting values
`noestimate`	do not fit the model; show starting values instead
`estmetric`	show parameter estimates in the estimation metric
`dnumerical`	use numerical derivative techniques
`coeflegend`	display legend instead of statistics

intmethod	Description
`mvaghermite`	mean–variance adaptive Gauss–Hermite quadrature; the default
`mcaghermite`	mode-curvature adaptive Gauss–Hermite quadrature
`ghermite`	nonadaptive Gauss–Hermite quadrature

`bootstrap`, `by`, `collect`, `jackknife`, `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**.
`startvalues()`, `noestimate`, `estmetric`, `dnumerical`, and `coeflegend` do not appear in the dialog box.
See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

`group(`*varname*`)` specifies that the model be fit separately for the different values of *varname*; see [IRT] **irt, group()** for details.

Model

`cns(`*spec*`)` constrains item parameters to a fixed value or constrains two or more parameters to be equal; see [IRT] **irt constraints** for details.

`listwise` handles missing values through listwise deletion, which means that the entire observation is omitted from the estimation sample if any of the items are missing for that observation. By default, all nonmissing items in an observation are included in the likelihood calculation; only missing items are excluded.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory (`oim`), that are robust to some kinds of misspecification (`robust`), that allow for intragroup correlation (`cluster` *clustvar*), and that use bootstrap or jackknife methods (`bootstrap`, `jackknife`); see [R] ***vce_option***.

Reporting

`level(#)`; see [R] **Estimation options**.

`notable` suppresses the estimation table, either at estimation or upon replay.

`noheader` suppresses the output header, either at estimation or upon replay.

display_options: `noci`, `nopvalues`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Integration

`intmethod(`*intmethod*`)` specifies the integration method to be used for computing the log likelihood. `mvaghermite` performs mean and variance adaptive Gauss–Hermite quadrature; `mcaghermite` performs mode and curvature adaptive Gauss–Hermite quadrature; and `ghermite` performs nonadaptive Gauss–Hermite quadrature.

The default integration method is `mvaghermite`.

`intpoints(#)` sets the number of integration points for quadrature. The default is `intpoints(7)`, which means that seven quadrature points are used to compute the log likelihood.

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases with the number of integration points.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, `nonrtolerance`, and `from(`*init_specs*`)`; see [R] **Maximize**. Those that require special mention for `irt` are listed below.

`from()` accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with irt but are not shown in the dialog box:

startvalues() specifies how starting values are to be computed. Starting values specified in from() override the computed starting values.

startvalues(zero) specifies that all starting values be set to 0. This option is typically useful only when specified with the from() option.

startvalues(constantonly) builds on startvalues(zero) by fitting a constant-only model for each response to obtain estimates of intercept and cutpoint parameters.

startvalues(fixedonly) builds on startvalues(constantonly) by fitting a full fixed-effects model for each response variable to obtain estimates of coefficients along with intercept and cutpoint parameters. You can also add suboption iterate(#) to limit the number of iterations irt allows for fitting the fixed-effects model.

startvalues(ivloadings) 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. This is the default behavior.

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 coeflegend style of output. An important use of this option is before you have modified starting values at all; you can type the following:

```
. irt ..., ... noestimate
. matrix b = e(b)
. ... (modify elements of b) ...
. irt ..., ... from(b)
```

estmetric displays parameter estimates in the slope-intercept metric that is used for estimation.

dnumerical specifies that during optimization, the gradient vector and Hessian matrix be computed using numerical techniques instead of analytical formulas. By default, irt uses analytical formulas for computing the gradient and Hessian for all integration methods.

coeflegend; see [R] **Estimation options**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Overview

The following discussion is about how to use irt to fit 2PL models to binary items. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

In the 2PL model, item responses are typically of the form yes or no, correct or incorrect, agree or disagree, etc. Items are assumed to vary in discrimination and difficulty. The probability of person j providing a positive answer to item i is given by

$$\Pr(Y_{ij} = 1|\theta_j) = \frac{\exp\{a_i(\theta_j - b_i)\}}{1 + \exp\{a_i(\theta_j - b_i)\}} \qquad \theta_j \sim N(0,1)$$

where a_i represents the discrimination of item i, b_i represents the difficulty of item i, and θ_j is the latent trait of person j.

The 2PL model was proposed by Birnbaum (1968). An earlier two-parameter model using a probit link was developed by Lord (1952).

See *Item response theory* in [BAYES] **bayesmh** and Balov (2016) for examples demonstrating Bayesian estimation of the 2PL model.

Example 1: Fitting a 2PL model

To illustrate the 2PL model, we use an abridged version of the mathematics and science data from De Boeck and Wilson (2004). Student responses to test items are coded 1 for correct and 0 for incorrect. Here we list the first five observations.

```
. use https://www.stata-press.com/data/r18/masc1
(Data from De Boeck & Wilson (2004))
. list in 1/5
```

	q1	q2	q3	q4	q5	q6	q7	q8	q9
1.	1	1	1	0	0	0	0	1	0
2.	0	0	1	0	0	0	0	1	1
3.	0	0	0	1	0	0	1	0	0
4.	0	0	1	0	0	0	0	0	1
5.	0	1	1	0	0	0	0	1	0

Looking across the rows, we see that the first student correctly answered items q1, q2, q3, and q8, the second student correctly answered items q3, q8, and q9, and so on.

We fit a 2PL model to binary items q1–q9 as follows:

```
. irt 2pl q1-q9
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -4275.6606
Iteration 1:  Log likelihood = -4269.7861
Iteration 2:  Log likelihood = -4269.7825
Iteration 3:  Log likelihood = -4269.7825
Fitting full model:
Iteration 0:  Log likelihood = -4146.9386
Iteration 1:  Log likelihood = -4119.3568
Iteration 2:  Log likelihood = -4118.4716
Iteration 3:  Log likelihood = -4118.4697
Iteration 4:  Log likelihood = -4118.4697
Two-parameter logistic model                    Number of obs = 800
Log likelihood = -4118.4697
```

		Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1							
	Discrim	1.615292	.2436467	6.63	0.000	1.137754	2.092831
	Diff	-.4745635	.074638	-6.36	0.000	-.6208513	-.3282757
q2							
	Discrim	.6576171	.1161756	5.66	0.000	.4299171	.885317
	Diff	-.1513023	.1202807	-1.26	0.208	-.3870481	.0844435
q3							
	Discrim	.9245051	.1569806	5.89	0.000	.6168289	1.232181
	Diff	-1.70918	.242266	-7.05	0.000	-2.184012	-1.234347
q4							
	Discrim	.8186403	.1284832	6.37	0.000	.5668179	1.070463
	Diff	.3296791	.1076105	3.06	0.002	.1187663	.5405919
q5							
	Discrim	.8956621	.1535128	5.83	0.000	.5947825	1.196542
	Diff	1.591164	.2325918	6.84	0.000	1.135293	2.047036
q6							
	Discrim	.9828441	.147888	6.65	0.000	.6929889	1.272699
	Diff	.622954	.1114902	5.59	0.000	.4044373	.8414708
q7							
	Discrim	.3556064	.1113146	3.19	0.001	.1374337	.5737791
	Diff	2.840278	.8717471	3.26	0.001	1.131685	4.548871
q8							
	Discrim	1.399926	.233963	5.98	0.000	.9413668	1.858485
	Diff	-1.714416	.1925531	-8.90	0.000	-2.091814	-1.337019
q9							
	Discrim	.6378452	.1223972	5.21	0.000	.3979512	.8777392
	Diff	-1.508254	.2787386	-5.41	0.000	-2.054571	-.9619361

In the 2PL model, each test item has its own parameter estimates for discrimination and difficulty.

In the following, we use `estat report` to replay the table of estimated IRT parameters and control how the output is reported. We include the `byparm` option, which arranges the output by parameter rather than by item, and the `sort(a)` option, which displays the items in an ascending order of discrimination. This makes it easy to see that item `q7` is least discriminating (`Discrim` = 0.36) and item `q1` is most discriminating (`Discrim` = 1.62).

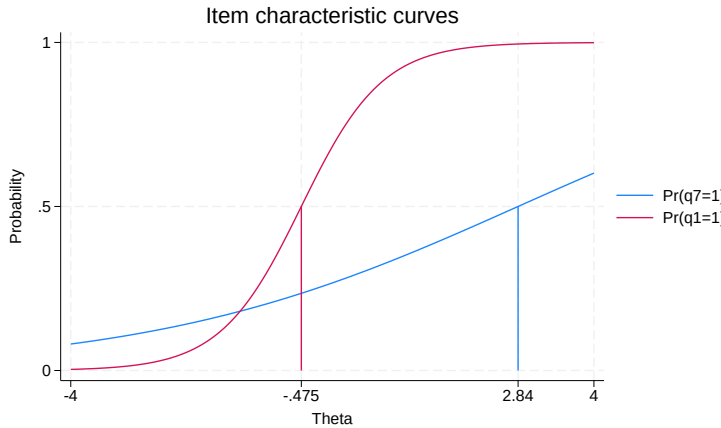
```
. estat report, byparm sort(a)
Two-parameter logistic model                    Number of obs = 800
Log likelihood = -4118.4697
```

		Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim							
	q7	.3556064	.1113146	3.19	0.001	.1374337	.5737791
	q9	.6378452	.1223972	5.21	0.000	.3979512	.8777392
	q2	.6576171	.1161756	5.66	0.000	.4299171	.885317
	q4	.8186403	.1284832	6.37	0.000	.5668179	1.070463
	q5	.8956621	.1535128	5.83	0.000	.5947825	1.196542
	q3	.9245051	.1569806	5.89	0.000	.6168289	1.232181
	q6	.9828441	.147888	6.65	0.000	.6929889	1.272699
	q8	1.399926	.233963	5.98	0.000	.9413668	1.858485
	q1	1.615292	.2436467	6.63	0.000	1.137754	2.092831
Diff							
	q7	2.840278	.8717471	3.26	0.001	1.131685	4.548871
	q9	-1.508254	.2787386	-5.41	0.000	-2.054571	-.9619361
	q2	-.1513023	.1202807	-1.26	0.208	-.3870481	.0844435
	q4	.3296791	.1076105	3.06	0.002	.1187663	.5405919
	q5	1.591164	.2325918	6.84	0.000	1.135293	2.047036
	q3	-1.70918	.242266	-7.05	0.000	-2.184012	-1.234347
	q6	.622954	.1114902	5.59	0.000	.4044373	.8414708
	q8	-1.714416	.1925531	-8.90	0.000	-2.091814	-1.337019
	q1	-.4745635	.074638	-6.36	0.000	-.6208513	-.3282757

The estimates of the difficulty parameter correspond to the point on the latent trait scale at which $\Pr(Y = 1|\theta) = 0.5$. Because we assume a zero mean for θ, an item is said to be relatively easy if its difficulty estimate is negative and relatively hard if its difficulty estimate is positive.

After `irt 2pl`, we can use `irtgraph icc` to plot the ICCs using the estimated 2PL parameters; see [IRT] **irtgraph icc**. To focus on the items with the highest and lowest discrimination, as shown by `estat report`, we plot only items `q7` and `q1`. We use option `blocation` to add vertical lines for item difficulties.

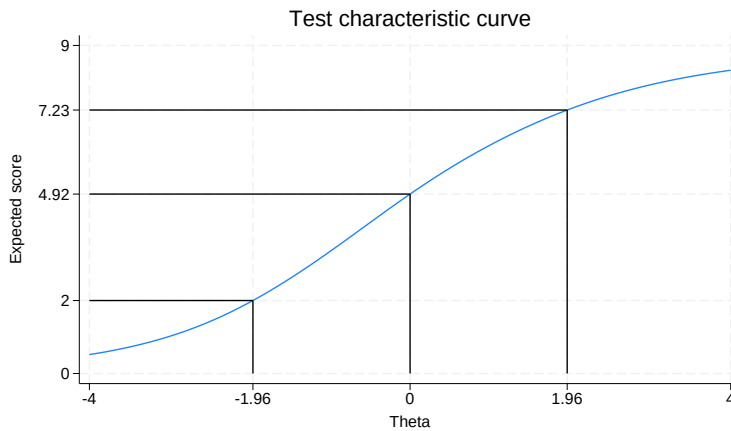
```
. irtgraph icc q7 q1, blocation
```



We chose to plot the ICC for items `q1` and `q7` to show that the estimated discrimination parameters give a sense of the slope of the ICC at the point where θ is equal to the estimated difficulty parameter. Given a high discrimination of item `q1`, its ICC has the steepest slope at its estimated difficulty parameter when compared with the slopes of the ICC of the other items at their estimated difficulty parameter. Likewise, the ICC for `q7` has the most gradual slope.

We use `irtgraph tcc` to plot the TCC using the estimated 2PL parameters; see [IRT] **irtgraph tcc**. For 9 binary items, it is clear that the total score ranges from 0 to 9. The `thetalines()` option plots the expected scores at the specified values for θ.

```
. irtgraph tcc, thetalines(-1.96 0 1.96)
```



This plot tells us what kind of scores we can expect from individuals with different levels of latent trait. For example, we can expect above-average individuals to score 4.92 or above. Actually, no one is expected to score 4.92 on a 9-item test, so a more realistic statement is that we expect above-average individuals to score above 4.

Using the 95% critical values from the standard normal distribution (-1.96 and 1.96), this plot also tells us that we can expect 95% of randomly selected people to score between 2 and 7.23. Again,

a more realistic statement is that we expect about 95% of randomly selected people to score from 2 to 7.

◁

Video example

Item response theory using Stata: Two-parameter logistic (2PL) models

Stored results

`irt 2pl` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(k_eq)`	number of equations in `e(b)`
`e(k_dv)`	number of dependent variables
`e(k_rc)`	number of covariances
`e(k_rs)`	number of variances
`e(irt_k_eq)`	number of IRT equations
`e(k_items1)`	number of items in first IRT equation
`e(ll)`	log likelihood
`e(N_clust)`	number of clusters
`e(N_groups)`	number of groups
`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)`	`gsem`
`e(cmd2)`	`irt`
`e(cmdline)`	command as typed
`e(model1)`	`2pl`
`e(items1)`	names of items in first IRT equation
`e(depvar)`	names of all item variables
`e(wtype)`	weight type
`e(wexp)`	weight expression
`e(title)`	title in estimation output
`e(clustvar)`	name of cluster variable
`e(groupvar)`	name of group variable
`e(family#)`	family for the #th *item*
`e(link#)`	link for the #th *item*
`e(intmethod)`	integration method
`e(vce)`	*vcetype* specified in `vce()`
`e(vcetype)`	title used to label Std. err.
`e(opt)`	type of optimization
`e(which)`	`max` or `min`; whether optimizer is to perform maximization or minimization
`e(method)`	estimation method: `ml`
`e(ml_method)`	type of `ml` method
`e(user)`	name of likelihood-evaluator program
`e(technique)`	maximization technique
`e(datasignature)`	the checksum
`e(datasignaturevars)`	variables used in calculation of checksum
`e(properties)`	`b V`
`e(estat_cmd)`	program used to implement `estat`
`e(predict)`	program used to implement `predict`
`e(covariates)`	list of covariates
`e(footnote)`	program used to implement the footnote display

Matrices

e(_N)	sample size for each item
e(b)	coefficient vector, slope-intercept parameterization
e(b_pclass)	parameter class
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance
e(groupvalue)	vector of group values in e(groupvar)
e(nobs)	vector with number of observations per group

Functions

e(sample)	marks estimation sample

In addition to the above, the following is stored in r():

Matrices

r(table)	matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in r() are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Let Y_{ij} represent the (yet to be observed) outcome for item i from person j, and let y_{ij} be the observed value of Y_{ij}. Without loss of generality, we will use the terms "correct" and "incorrect" in reference to the outcomes of Y_{ij}. Furthermore, we will refer to $y_{ij} = 1$ as correct and $y_{ij} = 0$ as incorrect.

Using the IRT parameterization, we see that the probability of person j with latent trait level θ_j providing a correct response to item i is given by

$$\Pr(Y_{ij} = 1|a_i, b_i, \theta_j) = \frac{\exp\{a_i(\theta_j - b_i)\}}{1 + \exp\{a_i(\theta_j - b_i)\}}$$

where a_i represents the discrimination of item i, and b_i represents the difficulty of item i. irt 2pl fits the model using the slope-intercept form, so the probability for providing a correct answer is parameterized as

$$\Pr(Y_{ij} = 1|\alpha_i, \beta_i, \theta_j) = \frac{\exp(\alpha_i\theta_j + \beta_i)}{1 + \exp(\alpha_i\theta_j + \beta_i)}$$

The transformation between these two parameterizations is

$$a_i = \alpha_i \qquad b_i = -\frac{\beta_i}{\alpha_i}$$

Let $p_{ij} = \Pr(Y_{ij} = 1|\alpha_i, \beta_i, \theta_j)$ and $q_{ij} = 1 - p_{ij}$. Conditional on θ_j, the item responses are assumed to be independent, so the conditional density for person j is given by

$$f(\mathbf{y}_j|\boldsymbol{B}, \theta_j) = \prod_{i=1}^{I} p_{ij}^{y_{ij}} q_{ij}^{1-y_{ij}}$$

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha_1, \ldots, \alpha_I, \beta_1, \ldots, \beta_I)$, and I is the number of items.

Missing items are skipped over in the above product by default. When the `listwise` option is specified, persons with any missing items are dropped from the estimation sample.

The likelihood for person j is computed by integrating out the latent variable from the joint density

$$L_j(\boldsymbol{B}) = \int_{-\infty}^{\infty} f(\mathbf{y}_j|\boldsymbol{B},\theta_j)\,\phi(\theta_j)\,d\theta_j$$

where $\phi(\cdot)$ is the density function for the standard normal distribution. The log likelihood for the estimation sample is simply the sum of the log likelihoods from the N persons in the estimation sample.

$$\log L(\boldsymbol{B}) = \sum_{j=1}^{N} \log L_j(\boldsymbol{B})$$

The integral in the formula for $L_j(\boldsymbol{B})$ is generally not tractable, so we must use numerical methods.

Models for multiple groups, Gauss–Hermite quadrature, and adaptive quadrature are documented in *Methods and formulas* of [IRT] **irt hybrid**.

References

Balov, N. 2016. Bayesian binary item response theory models using bayesmh. *The Stata Blog: Not Elsewhere Classified*. http://blog.stata.com/2016/01/18/bayesian-binary-item-response-theory-models-using-bayesmh/.

Birnbaum, A. 1968. Some latent trait models and their use in inferring an examinee's ability. In *Statistical Theories of Mental Test Scores*, ed. F. M. Lord and M. R. Novick, 395–479. Reading, MA: Addison–Wesley.

De Boeck, P., and M. Wilson, ed. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer.

Lord, F. M. 1952. *A Theory of Test Scores*. Iowa City, IA: Psychometric Society.

Also see

[IRT] **irt 2pl postestimation** — Postestimation tools for irt 2pl

[IRT] **irt** — Introduction to IRT models

[IRT] **irt 1pl** — One-parameter logistic model

[IRT] **irt 3pl** — Three-parameter logistic model

[IRT] **irt constraints** — Specifying constraints

[SEM] **Example 29g** — Two-parameter logistic IRT model

[SEM] **gsem** — Generalized structural equation model estimation command

[SVY] **svy estimation** — Estimation commands for survey data

[U] **20 Estimation and postestimation commands**