STATA ITEM RESPONSE THEORY REFERENCE MANUAL
RELEASE 19

A Stata Press Publication
StataCorp LLC
College Station, Texas

Version 19

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845

ISBN-10: 1-59718-432-2
ISBN-13: 978-1-59718-432-8

The suggested citation for this software is

> StataCorp. 2025. *Stata 19*. Statistical software. StataCorp LLC.

The suggested citation for this manual is

> StataCorp. 2025. *Stata 19 Item Response Theory Reference Manual*. College Station, TX: Stata Press.

Contents

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*
[ADAPT]	*Stata Adaptive Designs: Group Sequential Trials Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[BMA]	*Stata Bayesian Model Averaging Reference Manual*
[CAUSAL]	*Stata Causal Inference and Treatment-Effects Estimation Reference Manual*
[CM]	*Stata Choice Models Reference Manual*
[D]	*Stata Data Management Reference Manual*
[DSGE]	*Stata Dynamic Stochastic General Equilibrium Models Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[H2OML]	*Machine Learning in Stata Using H2O: Ensemble Decision Trees Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[LASSO]	*Stata Lasso Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[META]	*Stata Meta-Analysis Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power, Precision, and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[RPT]	*Stata Reporting Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TABLES]	*Stata Customizable Tables and Collected Results Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[I]	*Stata Index*
[M]	*Mata Reference Manual*

Description Remarks and examples References Also see

Description

Item response theory (IRT) is used in the design, analysis, scoring, and comparison of tests and similar instruments whose purpose is to measure unobservable characteristics of the respondents. This entry discusses some fundamental and theoretical aspects of IRT and illustrates these with worked examples.

The entries that follow describe how you can use the `irt` suite of commands to fit a variety of IRT models and to evaluate the results. The commands for fitting models can be grouped by the type of responses you are modeling.

Binary response models

`irt 1pl`	One-parameter logistic model
`irt 2pl`	Two-parameter logistic model
`irt 3pl`	Three-parameter logistic model

Categorical response models

`irt grm`	Graded response model
`irt nrm`	Nominal response model
`irt pcm`	Partial credit model
`irt rsm`	Rating scale model

Multiple IRT models combined

`irt hybrid`	Hybrid IRT models

These models can allow for differences across groups in the population.

Multiple-group IRT models

`irt, group()`	IRT models for multiple groups

Constraints can be applied when fitting any IRT model, and they are particularly useful for constraining parameters across groups in multiple-group models.

Constraints

`irt constraints`	Specifying constraints

After fitting any IRT model, results can be reported, interpreted, and evaluated using postestimation commands.

IRT graphs

`irtgraph icc`	Item characteristic curve plot
`irtgraph tcc`	Test characteristic curve plot
`irtgraph iif`	Item information function plot
`irtgraph tif`	Test information function plot

IRT reports

`estat report`	Report estimated IRT parameters
`estat greport`	Report estimated group IRT parameters

Model-specific postestimation overview

`irt 1pl postestimation`	Postestimation tools for irt 1pl
`irt 2pl postestimation`	Postestimation tools for irt 2pl
`irt 3pl postestimation`	Postestimation tools for irt 3pl
`irt grm postestimation`	Postestimation tools for irt grm
`irt nrm postestimation`	Postestimation tools for irt nrm
`irt pcm postestimation`	Postestimation tools for irt pcm
`irt rsm postestimation`	Postestimation tools for irt rsm
`irt hybrid postestimation`	Postestimation tools for irt hybrid
`irt, group() postestimation`	Postestimation tools for group IRT

Differential item functioning (DIF) occurs when items that are intended to measure a trait are unfair, favoring one group of individuals over another. DIF can be evaluated by fitting a multiple-group IRT model using `irt, group()` or by using a logistic regression or Mantel–Haenszel DIF test.

Differential item functioning

`DIF`	Introduction to differential item functioning
`diflogistic`	Logistic regression DIF
`difmh`	Mantel–Haenszel DIF

Remarks and examples

Researchers are often interested in studying abilities, personality traits, and other unobservable characteristics. Throughout this manual, we most often refer to the unobserved characteristic of interest as the latent trait, but we will sometimes also use the term ability.

Latent traits cannot be measured directly, because they are unobservable, but they can be quantified with an instrument. An instrument is simply a collection of items designed to measure a person's level of the latent trait. For example, a researcher interested in measuring mathematical ability (latent trait) may design a test (instrument) consisting of 100 questions (items).

When designing the instrument or analyzing data from the instrument, the researcher is interested in how each individual item relates to the trait and how the group of items as a whole relates to this trait. IRT models allow us to study these relationships.

IRT models are used extensively in the study of cognitive and personality traits, health outcomes, and in the development of item banks and computerized adaptive testing. Some examples of applied work include measuring computer anxiety in grade school children (King and Bond 1996), assessing physical functioning in adults with HIV (Wu et al. 1997), and measuring the degree of public policy involvement of nutritional professionals (Boardley, Fox, and Robinson 1999).

The bulk of the theoretical work in IRT comes from the fields of psychometrics and educational measurement with key early contributions from Rasch (1960), Birnbaum (1968), Wright and Stone (1979), and Lord (1980). Some good introductory IRT reading includes Hambleton, Swaminathan, and Rogers (1991), McDonald (1999), Embretson and Reise (2000), Bond and Fox (2015), and de Ayala (2022). More advanced treatments are presented, for example, in Fischer and Molenaar (1995), van der Linden and Hambleton (1997), Baker and Kim (2004), and De Boeck and Wilson (2004). Raykov and Marcoulides (2018) provide a comprehensive treatment of IRT using Stata.

Benjamin Drake Wright (1926–2015) was born in Wilkes-Barre, Pennsylvania. Wright joined the US Navy in 1944 and went on to study physics at Cornell University. He interned with American physicist Charles H. Townes, and after joining the physics department at the University of Chicago, he became Robert S. Mulliken's research assistant.

His interests began to shift, and in 1957 he obtained a PhD in the philosophy of human development. When the University of Chicago received an IBM computer, Wright wrote a program for factor analysis and regression. While performing factor analyses for a market research firm, Wright became discomforted by the inconsistency of the results.

In 1960, psychometrician Georg Rasch gave a series of lectures on his measurement models at the University of Chicago, and Wright was won over by their stability. Together with Bruce Choppin, he wrote computer programs that would fit the Rasch measurement models. His advocacy in these models is reflected in his cofounding of the Rasch Measurement Social Interest Group, as part of the American Education Research Association (AERA), and the Institute for Objective Measurement, which publishes the *Journal of Applied Measurement* on a quarterly basis. Wright also developed a type of map for presenting the overall performance levels of students; this KIDMAP concept was implemented first by the Los Angeles Independent County School District in the 1980s and later by the Australian Council for Educational Research.

For his many contributions to measurement, which spanned multiple fields, Wright was honored with two conferences celebrating his work.

Frederic M. Lord (1912–2000) was born in Hanover, New Hampshire. He obtained a master's degree in educational psychology from the University of Minnesota and a PhD in psychology from Princeton University. In 1949, he became the director of statistical analysis for the Education Testing Service (ETS), where he would work for 33 years.

Lord devised models to categorize test questions based on difficulty and thus laid the foundation for item response theory. His work with the ETS had impacts on the Law School Admissions Test, the test of English as a Foreign Language, and the Graduate Record exam. Additionally, he coauthored a book with Melvin R. Novick on test theory, which was an expansion of his dissertation. His dissertation alone made a lasting impact on psychometrics, as did his other publications.

In 2000, the ETS created the Frederic M. Lord Chair in Measurement and Statistics in his honor. Because of his pioneering contributions, Lord is regarded as the "Father of Modern Testing".

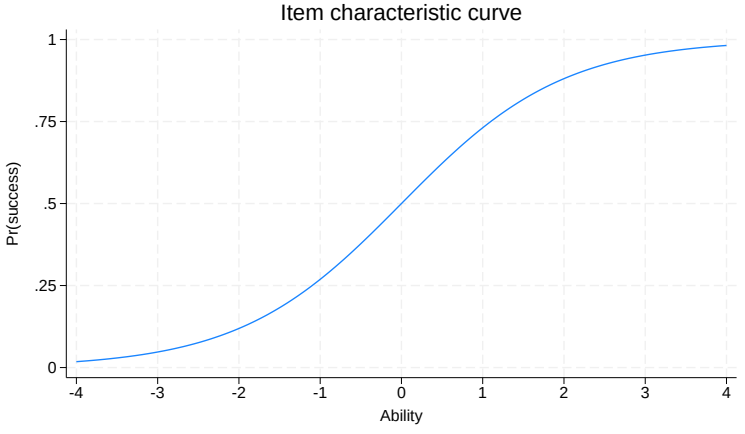
Allan Birnbaum (1923–1976) was born in San Francisco, California. He completed a premedical program prior to obtaining his PhD in mathematical statistics from Columbia University in 1954. There, among other projects, he worked on developing statistical methods applicable to the social sciences. In 1959, he joined the faculty of New York University, where he would teach statistics.

He published a total of 41 papers, but the paper published in 1962 stands out as his most significant contribution to the field of statistical theory. In this publication, he advocated for the likelihood principle, providing proof that the same inference can be made across two experiments that provided proportional likelihood functions. His approach departed from that of Abraham Wald and Erich Leo Lehmann, who influenced his dissertation. Although met simultaneously with appraise and opposition, his work had an impact on meta-analysis and predictions with missing data. Notably, renowned statistician Leonard Jimmie Savage regarded Birnbaum's work on the likelihood principle as highly influential in the field of statistics.

Birnbaum also published in the areas of classification and discrimination, and he applied his medical background to research on experimental genetics. He held faculty positions at Stanford University, New York University, and Cambridge University. The last position he held was chair of statistics at City, University of London. He was honored with election to fellowship by the American Association for the Advancement of Sciences, the American Statistical Association, and the Institute of Mathematical Statistics.

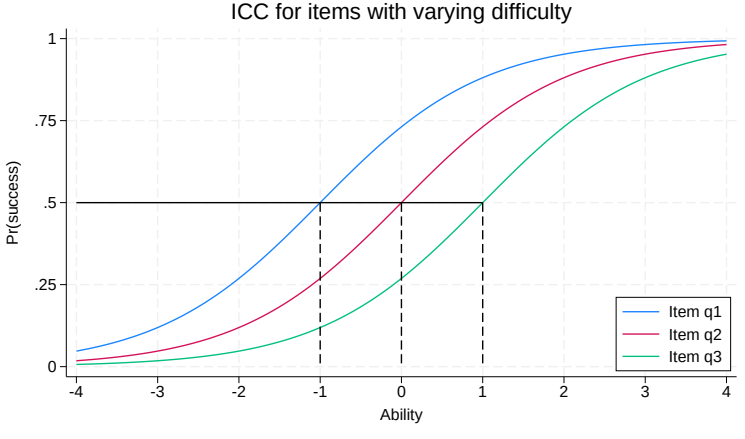
Birnbaum is remembered as a deep thinker and dedicated father.

The main concept in IRT is the item characteristic curve (ICC). The ICC describes the probability that a person "succeeds" on a given item (individual test question). In the following graph, we can see an ICC for one item intended to measure ability. Notice that the probability of this broadly defined success increases as ability increases.



ICCs will be different for different items. The probability of success on an item is a function of both the level of the latent trait and the properties of the item. The latent trait is commonly denoted by θ. The value of θ for a given person is called the person location. The item properties are parameters, commonly known as difficulty and discrimination, that are estimated in the IRT model.

The difficulty parameter, or item location, commonly denoted by b, represents the location of an item on the ability scale. For example, the following graph plots the ICC for items `q1`, `q2`, and `q3`, with difficulty parameters -1, 0, and 1, respectively.



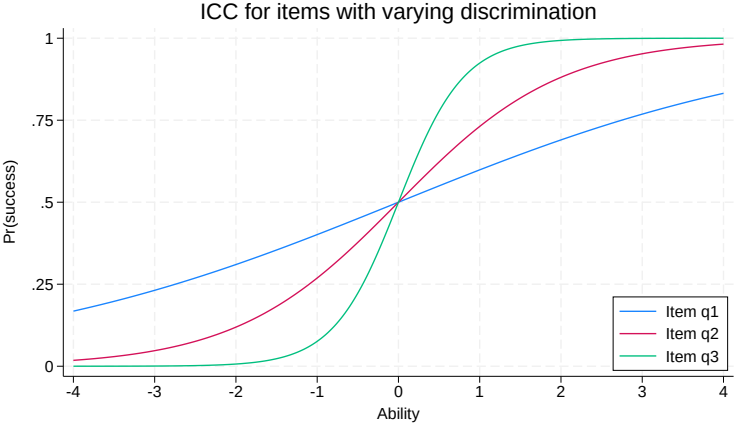
Item `q1` is the least difficult, and item `q3` is the most difficult. Notice that the change in difficulty shifts the ICC along the ability scale (that is, the horizontal axis or x axis). The probability of success on item `q1` is higher than the probability of success for the other two items at any ability level. We can say item `q1` is less difficult than the others because a person would need only an ability level greater than -1

on this ability scale to be expected to succeed on item q1. On the other hand, a person would need an ability level above 0 to be expected to succeed on item q2 and an ability level above 1 to be expected to succeed on item q3.

In designing an instrument intended to differentiate between all levels of a latent trait, a researcher should try to have items with difficulties spread across the full range of the trait.

The second item parameter, discrimination, is related to the slope of the ICC. Discrimination is commonly denoted by a. This item parameter tells us how fast the probability of success changes with ability near the item difficulty. An item with a large discrimination value has a high correlation between the latent trait and the probability of success on that item. In other words, an item with a large discrimination parameter can distinguish better between low and high levels of the latent trait.

In the graph above, all three items have the same discrimination. In the graph below, all three items have the same difficulty, but they have different discrimination values. A highly discriminating item differentiates better, around its difficulty value, between persons of similar levels of the latent trait.



Imagine two persons, one with ability just below zero, and the other with ability just above zero. According to the ICC for item q1, these persons would have a similar probability of success on this item. According to the ICC for item q3, the person with the higher ability level would have a substantially higher probability of success on this item.

Using an IRT model, we can estimate the discrimination and difficulty parameters, a and b, for each item on an instrument designed to measure a particular latent trait. Throughout this manual, we assume that a single latent trait is sufficient to explain a person's response behavior on the group of items. More technically, we assume a unidimensional latent space. We also assume that after we condition on ability, a person's responses to an item are independent of his or her responses to other items. This is called a conditional independence or a local independence assumption.

We can now express a generic functional form of an ICC as

$$\Pr(\text{success}|a,b,\theta) = F\{a(\theta-b)\}$$

The difference term $(\theta - b)$ tells us that the probability of success is a function of the distance between item location and person location. When $\theta = b$, that is, when item difficulty is matched to a person's latent trait level, the individual is equally likely to pass or fail the item. When $\theta > b$, the individual is more likely to succeed than to fail. Because we can obtain the same distance with different choices of θ and b, we need to provide a metric for θ to identify the model. We do so by assuming $\theta \sim N(0,1)$,

which also puts the item difficulty parameter on the same scale as the standard normal distribution. With the standard normal scale, items with negative difficulties are considered to be relatively easy, and items with positive difficulties are considered to be relatively hard.

For any IRT model, we assume $F(\cdot)$ to be of correct functional form and increasing with the value of the latent trait. Because probabilities are bounded between 0 and 1, $F(\cdot)$ is usually a variation of a cumulative logistic distribution.

Through choices of $F(\cdot)$ and specification of certain constraints on the estimated parameters, we can fit a variety of different types of IRT models. Using the `irt` commands, we can fit IRT models to binary, ordinal, and nominal items. Below we demonstrate an IRT model with binary items and an IRT model with ordinal items. For additional information and examples of the models available for binary items, see [IRT] **irt 1pl**, [IRT] **irt 2pl**, and [IRT] **irt 3pl**. For models with ordinal items, see [IRT] **irt grm**, [IRT] **irt rsm**, and [IRT] **irt pcm**. For models with nominal items, see [IRT] **irt nrm**. Each of these models can allow parameters to differ across groups such as males and females or age categories; see [IRT] **irt, group( )**. In addition to fitting the models, we can better understand each item and its relationship to the latent trait through a variety of graphs, as demonstrated in the examples below.

The `irt` commands fit IRT models via maximum likelihood estimation. See *Item response theory* in [BAYES] **bayesmh** and Balov (2016) for examples of fitting IRT models to binary items using a Bayesian approach.

From a broader statistical perspective, IRT models can be viewed as extensions of (unidimensional) confirmatory factor analysis (CFA) models to binary and categorical outcomes and as special cases of generalized linear mixed-effects models; see chapter 1 in De Boeck and Wilson (2004) and chapter 3 in Skrondal and Rabe-Hesketh (2004) for a theoretical discussion and Zheng and Rabe-Hesketh (2007) for applied examples.

▷ Example 1: Binary IRT models

In this example, we present IRT analysis of binary data and highlight some postestimation features of `irt`. 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/r19/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.

Let's say the goal of the test is to assess students' mathematical ability and perhaps classify the students into groups, for example, gifted, average, and remedial. We could look at the total test score for each student, but the problem is that the total score depends on the composition of the test. If the test comprises easy items, most students will appear to be gifted, and if the test comprises hard items, most students

will be assigned to the remedial group. When the model fits the data, an attractive property of IRT is that, except for measurement error, parameter estimates are invariant; that is, examinee ability estimates are not test dependent, and item parameter estimates are not group dependent.

We fit a 1PL model to binary items q1–q9 as follows.

```
. irt 1pl 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 = -4153.3609
Iteration 1:  Log likelihood =  -4142.374
Iteration 2:  Log likelihood = -4142.3516
Iteration 3:  Log likelihood = -4142.3516
One-parameter logistic model                    Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
q1						
Diff	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2						
Diff	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q3						
Diff	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q4						
Diff	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q5						
Diff	1.652719	.1329494	12.43	0.000	1.392144	1.913295
q6						
Diff	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7						
Diff	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q8						
Diff	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q9						
Diff	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448

Looking at the output table, we see that the first row reports the estimate of the item discrimination parameter, labeled Discrim. In a 1PL model, this parameter is shared by all items. The estimate of 0.85 suggests the items are not particularly discriminating; that is, in the vicinity of a given difficulty estimate, any two students with distinct abilities would have similar predicted probabilities of responding correctly

to an item. The remaining rows report the estimates of the difficulty parameters, labeled `Diff`, for each item. The items appear to cover a wide range of the item difficulty spectrum, with item `q8` being the lowest ($\widehat{b}_8 = -2.41$) and item `q5` being the highest ($\widehat{b}_5 = 1.65$).

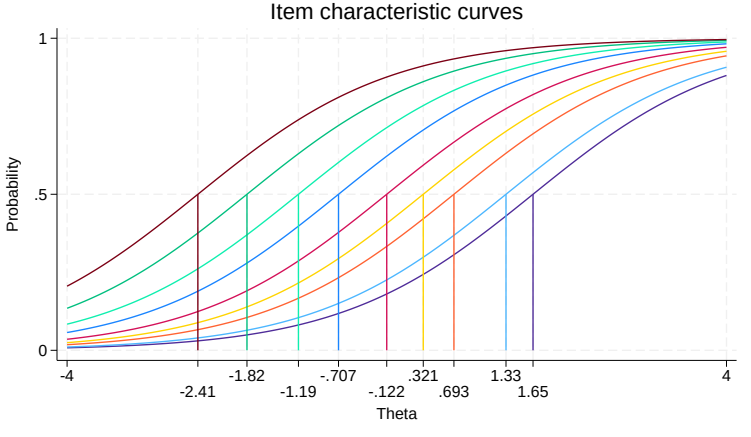
We use `estat report` to arrange the output in a particular sort order, which, in our example, makes it easy to see which items are easy and which are hard; see [IRT] **estat report** for details.

```
. estat report, sort(b) byparm
One-parameter logistic model                          Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
Diff						
q8	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q3	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q9	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448
q1	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q4	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q6	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q5	1.652719	.1329494	12.43	0.000	1.392144	1.913295

To visualize the item locations on the difficulty spectrum, we plot the ICCs for all items using `irtgraph icc`; see [IRT] **irtgraph icc** for details.

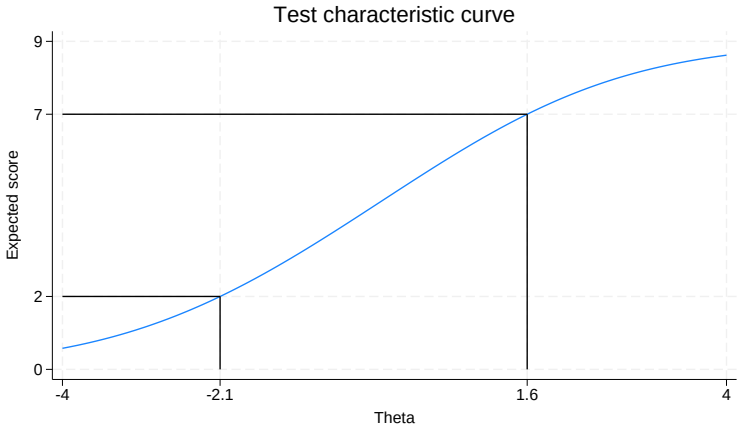
```
. irtgraph icc, blocation legend(off) xlabel(,alt)
```



The probabilities represent the expected scores for each item along the latent trait continuum. For the 1PL model, the midpoint probability for each item corresponds with the estimated difficulty parameter.

The sum of the probabilities gives us the expected score on the whole test. A plot of the expected score against the latent trait is called a test characteristic curve (TCC). Below we plot the TCC for our model using `irtgraph tcc`; see [IRT] **irtgraph tcc** for details. The `scorelines(2 7)` option specifies that droplines corresponding to the expected scores of 2 and 7 also be plotted. According to the estimated TCC, these expected scores correspond with the latent trait locations -2.1 and 1.6, respectively.

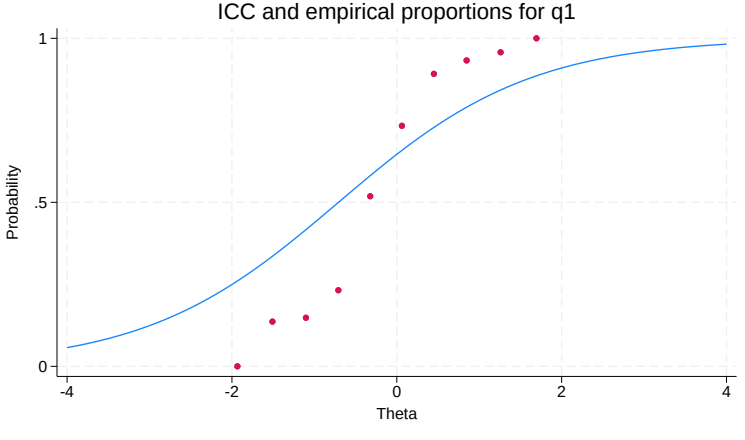
```
. irtgraph tcc, scorelines(2 7)
```



The invariance property of IRT holds only if the model fits the data. One informal method to check item fit is to superimpose empirical proportions on an ICC. If the predicted ICC follows closely the empirical trace line implied by the proportions, an item is assumed to have a satisfactory fit.

To calculate the empirical proportions, we predict the latent trait and collapse the items by the latent trait. We then call `irtgraph icc` with option `addplot()` to superimpose the proportions on the ICC.

```
. predict Theta, latent
(option ebmeans assumed)
(using 7 quadrature points)
. collapse q*, by(Theta)
. irtgraph icc q1, addplot(scatter q1 Theta)
> title("ICC and empirical proportions for q1")
```



We see that the fit of the ICC to the implied empirical trace line is poor. This is true for all items in the model. It is possible that a 2PL model may be more appropriate for this item. Before we fit a 2PL model, we store our estimates for later use.

```
. estimates store onep
```

To fit a 2PL model to the data, we type

```
. use https://www.stata-press.com/data/r19/masc1, clear
(Data from De Boeck & Wilson (2004))
. 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

Now each item has its own discrimination parameter that models the slope of the ICC for that item. In a 1PL model, the discrimination for all items was estimated to be 0.85. Looking at item q1 in the output table above, we see that its discrimination is estimated to be 1.62, which corresponds to a steeper slope and should result in a better item fit.

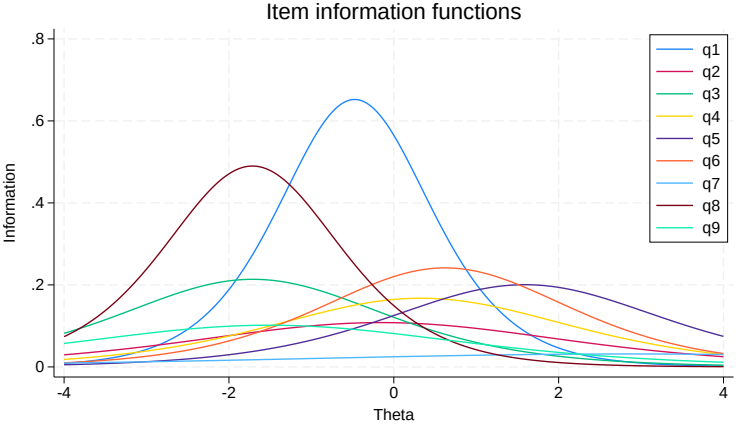
Because the 1PL model is nested in a 2PL model, we can perform a likelihood-ratio test to see which model is preferred.

```
. lrtest onep .
Likelihood-ratio test
Assumption: onep nested within .
 LR chi2(8) =  47.76
Prob > chi2 = 0.0000
```

The near-zero significance level favors the model that allows for a separate discrimination parameter for each item.

Continuing with the 2PL model, we can also plot the amount of information an item provides for estimating the latent trait. A plot of item information against the latent trait is called an item information function (IIF). We use `irtgraph iif` to obtain the IIFs for all items in the model; see [IRT] **irtgraph iif** for details.

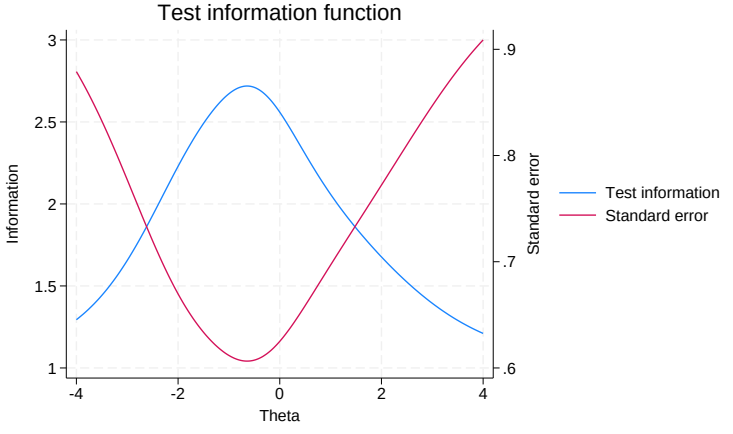
```
. irtgraph iif, legend(pos(1) ring(0) region(lcolor(black)))
```



For a 2PL model, IIFs are unimodal and symmetric, and each item provides the maximum amount of information at its estimated difficulty parameter. The height of an IIF and therefore the amount of information an item provides around the difficulty parameter is proportional to the item's estimated discrimination. Items q1 and q8 are most discriminating and have the steepest IIFs.

We can sum up all the IIFs to obtain a test information function (TIF). The TIF plot tells us how well the instrument can estimate person locations; see [IRT] **irtgraph tif** for details.

```
. irtgraph tif, se
```



The test provides maximum information for persons approximately located at $\theta = -0.5$. As we move away from that point in either direction, the standard error of the TIF increases, and the instrument provides less and less information about θ.

The TIF is useful in designing instruments targeted at obtaining precise estimates of a person's latent trait level at specified intervals. If our interest lies in identifying gifted and remedial students, we would like the instrument to be more precise at the extrema of the ability range. If we wish to have a similar precision of ability estimate across the entire ability range, we would like to see a relatively flat TIF. Because the TIF is a sum of IIFs, we can obtain the desired shape of the TIF by incorporating items targeted at a specified ability interval.

◁

The last binary model, not shown here, is a 3PL model. This model adds to the 2PL model by accommodating the possibility of guessing. We discuss this model in the [IRT] **irt 3pl** entry.

Example 2: Categorical IRT models

Categorical IRT models include models for ordered and unordered responses. Here we present a graded response model (GRM) for ordered responses.

The GRM is an extension of the 2PL model to categorical outcomes. To illustrate the model, we use the data from Zheng and Rabe-Hesketh (2007). `charity.dta` contains five survey questions, `ta1` through `ta5`, measuring faith and trust in charity organizations. Responses are strongly agree (0), agree (1), disagree (2), and strongly disagree (3). Higher scores indicate higher levels of distrust. Here we list the first five observations.

```
. use https://www.stata-press.com/data/r19/charity
(Data from Zheng & Rabe-Hesketh (2007))
. list in 1/5, nolabel
```

	ta1	ta2	ta3	ta4	ta5
1.	.	2	1	1	.
2.	0	0	0	0	0
3.	1	1	2	0	2
4.	1	2	2	0	1
5.	.	1	1	1	1

Looking across the first row, we see that the first respondent did not provide an answer to items `ta1` and `ta5`, answered 2 on item `ta2`, and answered 1 on items `ta3` and `ta4`. All `irt` commands exclude missing items for a given observation from the likelihood calculation but keep the nonmissing items for that observation. If you wish to remove the entire observation from the model, add the `listwise` option at estimation time.

We fit a GRM as follows:

```
. irt grm ta1-ta5
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -5559.6414
Iteration 1:  Log likelihood = -5473.9434
Iteration 2:  Log likelihood = -5467.4082
Iteration 3:  Log likelihood = -5467.3926
Iteration 4:  Log likelihood = -5467.3926
Fitting full model:
Iteration 0:  Log likelihood = -5271.0634
Iteration 1:  Log likelihood = -5162.5917
Iteration 2:  Log likelihood = -5159.2947
Iteration 3:  Log likelihood = -5159.2791
Iteration 4:  Log likelihood = -5159.2791
Graded response model                                   Number of obs = 945
Log likelihood = -5159.2791
```

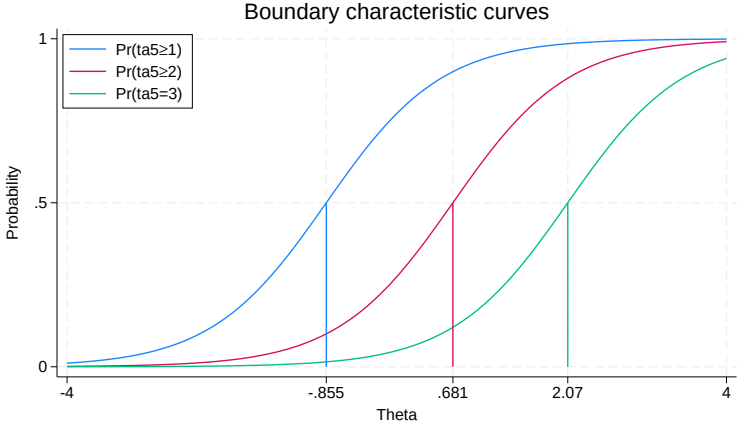
	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
ta1						
Discrim	.907542	.0955772	9.50	0.000	.7202142	1.09487
Diff						
>=1	-1.540098	.1639425			-1.861419	-1.218776
>=2	1.296135	.1427535			1.016343	1.575927
=3	3.305059	.3248468			2.668371	3.941747
ta2						
Discrim	.9434675	.0967483	9.75	0.000	.7538444	1.133091
Diff						
>=1	-1.661331	.167878			-1.990366	-1.332296
>=2	.0068314	.082222			-.1543208	.1679836
=3	2.531091	.2412513			2.058247	3.003935
ta3						
Discrim	1.734201	.1554383	11.16	0.000	1.429548	2.038855
Diff						
>=1	-1.080079	.0835119			-1.243759	-.9163983
>=2	1.016567	.0796635			.8604297	1.172705
=3	2.232606	.1497814			1.93904	2.526172
ta4						
Discrim	1.93344	.1857629	10.41	0.000	1.569351	2.297528
Diff						
>=1	-.3445057	.0578468			-.4578833	-.2311282
>=2	1.466254	.0983823			1.273428	1.65908
=3	2.418954	.162392			2.100672	2.737237
ta5						
Discrim	1.42753	.1263962	11.29	0.000	1.179798	1.675262
Diff						
>=1	-.8552358	.0833158			-1.018532	-.6919399
>=2	.6805315	.07469			.5341418	.8269211
=3	2.074243	.1538858			1.772632	2.375853

Because the GRM is derived in terms of cumulative probabilities, the estimated category difficulties represent a point at which a person with ability equal to a given difficulty has a 50% chance of responding in a category equal to or higher than the difficulty designates; see [IRT] **irt grm** for details. For example, looking at the estimated parameters of item `ta5`, we see that a person with $\theta = -0.86$ has a 50% chance

of answering 0 versus greater than or equal to 1, a person with $\theta = 0.68$ has a 50% chance of answering 0 or 1 versus greater than or equal to 2, and a person with $\theta = 2.07$ has a 50% chance of answering 0, 1, or 2 versus 3.

We can use `irtgraph icc` to plot these probabilities; here we show them for item `ta5` together with the estimated category difficulties. In a GRM, the midpoint probability for each category is located at the estimated category difficulty.

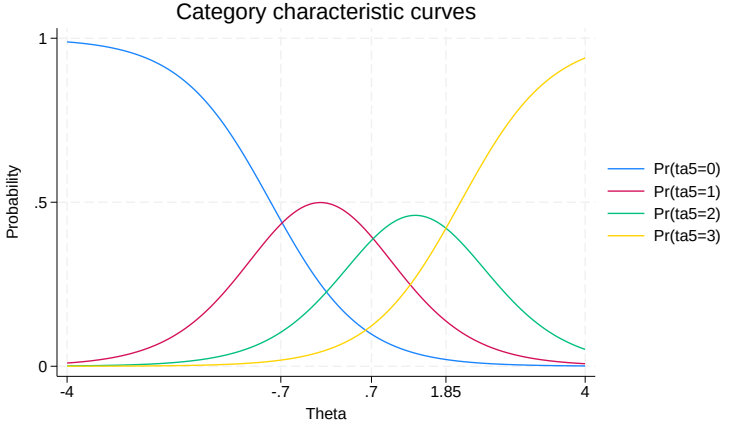
```
. irtgraph icc ta5, blocation legend(pos(11) ring(0) region(lcolor(black)))
```



When we plot characteristic curves for categorical items in ways reminiscent of ICCs for binary items, the resulting curves are called boundary characteristic curves (BCCs).

We can also plot the probabilities of respondents choosing exactly category k. For categorical items, the resulting curves are called category characteristic curves (CCCs). In fact, this is the default behavior of `irtgraph icc`.

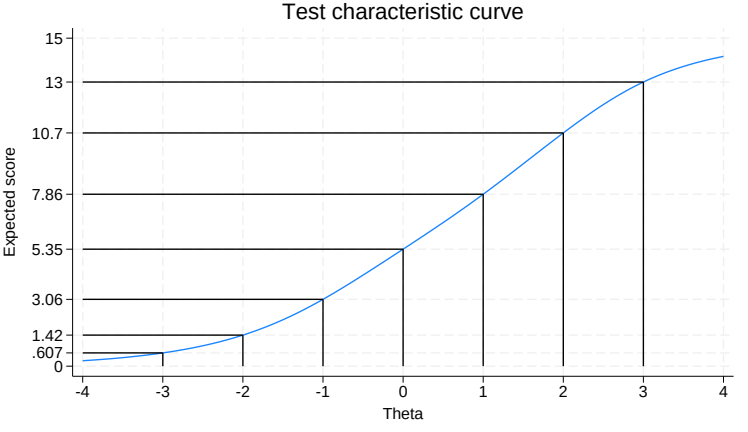
```
. irtgraph icc ta5, xlabel(-4 -.7 .7 1.85 4)
```



The points where the adjacent categories cross represent transitions from one category to the next. Thus, respondents with low levels of distrust, below approximately $\theta = -0.7$, are most likely to choose the first category on item `ta5` (strongly agree), respondents located approximately between -0.7 and 0.7 are most likely to choose the second category on item `ta5` (agree), and so on.

As in the first example, we can plot the test characteristic function for the whole instrument.

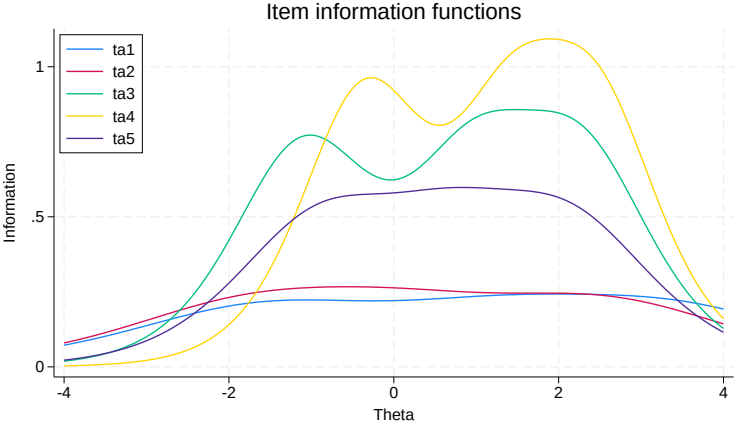
```
. irtgraph tcc, thetalines(-3/3)
```



Because we have 5 items, each with a minimum score of 0 and a maximum score of 3, the expected score ranges from 0 to 15. We also asked `irtgraph icc` to plot the expected scores for different values of θ. For respondents located at $\theta = -3$ and below, the expected score is less than 1, which means those respondents are most likely to choose the answer coded 0 on each and every item.

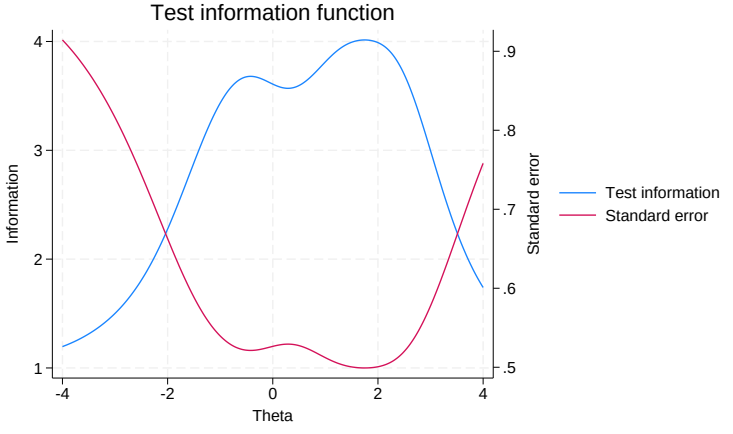
For categorical items, the item information function is no longer unimodal or symmetric, because each category contributes its own information, which may peak over a different ability range. We see this in the graph below.

```
. irtgraph iif, legend(pos(11) ring(0) region(lcolor(black)))
```



Because the test information function is the sum of the individual IIFs, its plot will also exhibit peaks and valleys.

```
. irtgraph tif, se
```



In the above example, we presented the GRM. The `irt` command also supports other models for categorical responses; see [IRT] **irt nrm** for a discussion of the nominal response model (NRM), [IRT] **irt pcm** for a discussion of the partial credit model (PCM), and [IRT] **irt rsm** for a discussion of the rating scale model (RSM).

In addition to binary and categorical IRT models, the `irt` command allows you to apply different models to subsets of items and perform a single calibration for the whole instrument. We call such models hybrid IRT models; see [IRT] **irt hybrid** for a further discussion and examples.

References

Baker, F. B., and S.-H. Kim. 2004. *Item Response Theory: Parameter Estimation Techniques*. 2nd ed, revised and expanded. Boca Raton, FL: CRC Press.

Balov, N. 2016. Bayesian binary item response theory models using bayesmh. *The Stata Blog: Not Elsewhere Classified*. https://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*, edited by F. M. Lord and M. R. Novick, 395–479. Reading, MA: Addison–Wesley.

Boardley, D., C. M. Fox, and K. L. Robinson. 1999. Public policy involvement of nutrition professionals. *Journal of Nutrition Education* 31: 248–254. https://doi.org/10.1016/S0022-3182(99)70460-7.

Bond, T. G., and C. M. Fox. 2015. *Applying the Rasch Model: Fundamental Measurement in the Human Sciences*. 3rd ed. New York: Routledge.

de Ayala, R. J. 2022. *The Theory and Practice of Item Response Theory*. 2nd ed. New York: Guilford Press.

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Embretson, S. E., and S. P. Reise. 2000. *Item Response Theory for Psychologists*. Mahwah, NJ: Lawrence Erlbaum.

Fischer, G. H., and I. W. Molenaar, eds. 1995. *Rasch Models: Foundations, Recent Developments, and Applications*. New York: Springer.

Hambleton, R. K., H. Swaminathan, and H. J. Rogers. 1991. *Fundamentals of Item Response Theory*. Newbury Park, CA: Sage.

King, J., and T. G. Bond. 1996. A Rasch analysis of a measure of computer anxiety. *Journal of Educational Computing Research* 14: 49–65. https://doi.org/10.2190/URRN-X4N9-V74C-U621.

Kondratek, B. 2022. uirt: A command for unidimensional IRT modeling. *Stata Journal* 22: 243–268.

Lord, F. M. 1980. *Applications of Item Response Theory to Practical Testing Problems*. Mahwah, NJ: Lawrence Erlbaum.

McDonald, R. P. 1999. *Test Theory: A Unified Treatment*. Mahwah, NJ: Lawrence Erlbaum.

Perrot, B., E. Bataille, and J.-B. Hardouin. 2018. validscale: A command to validate measurement scales. *Stata Journal* 18: 29–50.

Raciborski, R. 2015. Spotlight on irt. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2015/07/31/spotlight-on-irt/.

Rasch, G. 1960. *Probabilistic Models for Some Intelligence and Attainment Tests*. Copenhagen: Danish Institute of Educational Research.

Raykov, T., and G. A. Marcoulides. 2018. *A Course in Item Response Theory and Modeling with Stata*. College Station, TX: Stata Press.

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.

Wright, B. D., and M. H. Stone. 1979. *Best Test Design: Rasch Measurement*. Chicago: MESA Press.

Wu, A. W., R. D. Hays, S. Kelly, F. Malitz, and S. A. Bozzette. 1997. Applications of the Medical Outcomes Study health-related quality of life measures in HIV/AIDS. *Quality of Life Research* 6: 531–554. https://doi.org/10.1023/A:1018460132567.

Zheng, X., and S. Rabe-Hesketh. 2007. Estimating parameters of dichotomous and ordinal item response models with gllamm. *Stata Journal* 7: 313–333.

Also see

[IRT] **Glossary**

[IRT] **DIF** — Introduction to differential item functioning

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

Description Remarks and examples Reference Also see

Description

The IRT Control Panel allows you to perform a complete IRT analysis. From the Control Panel, you can fit IRT models, create customized reports of the results, and produce graphs of item characteristic curves (ICCs), category characteristic curves (CCCs), test characteristic curves (TCCs), item information functions (IIFs), and test information functions (TIFs).

Remarks and examples

You can perform IRT analyses using the `irt` commands, `estat report`, `estat greport`, and the `irtgraph` commands, or you can perform complete analyses interactively using the IRT Control Panel. Any analysis that you can perform with the `irt` commands, you can also perform from the Control Panel. This includes customizing graphs and reports and testing for differential item functioning.

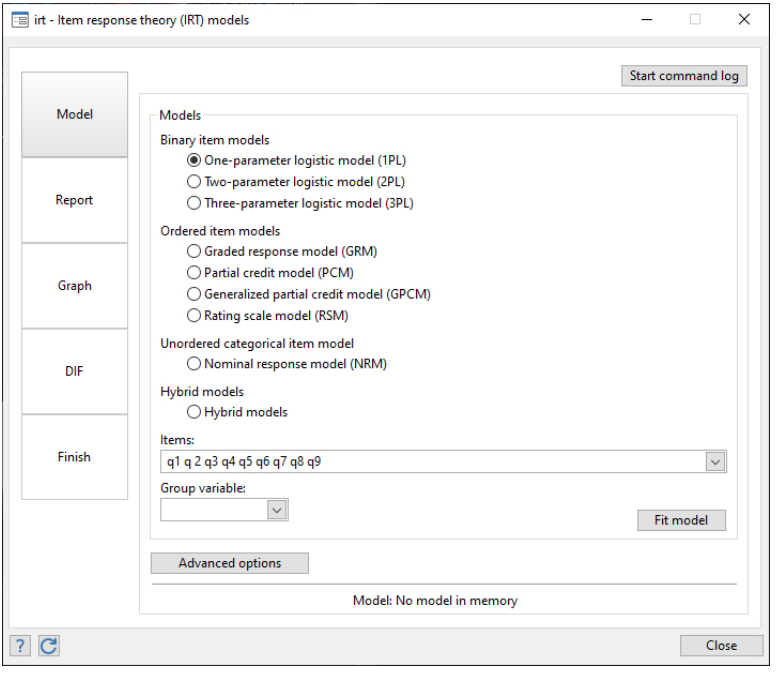
To demonstrate the IRT Control Panel, we will work example 1 of [IRT] **irt**. We open the abridged version of the mathematics and science data from De Boeck and Wilson (2004) by typing

```
. use https://www.stata-press.com/data/r19/masc1
```

in the Command window. This dataset contains 9 binary items, `q1` through `q9`, coded as 1 for correct and 0 for incorrect.

To open the IRT Control Panel, we select **Statistics** > **IRT (item response theory)** from the Stata menu.

The Control Panel opens to the *Model* tab, where we select the type of IRT model we wish to fit. Our example begins by fitting a one-parameter logistic (1PL) model to all nine items in the dataset, so we choose the *One-parameter logistic model (1PL)* radio button. Then, we select items q1 through q9 in the *Items* control.

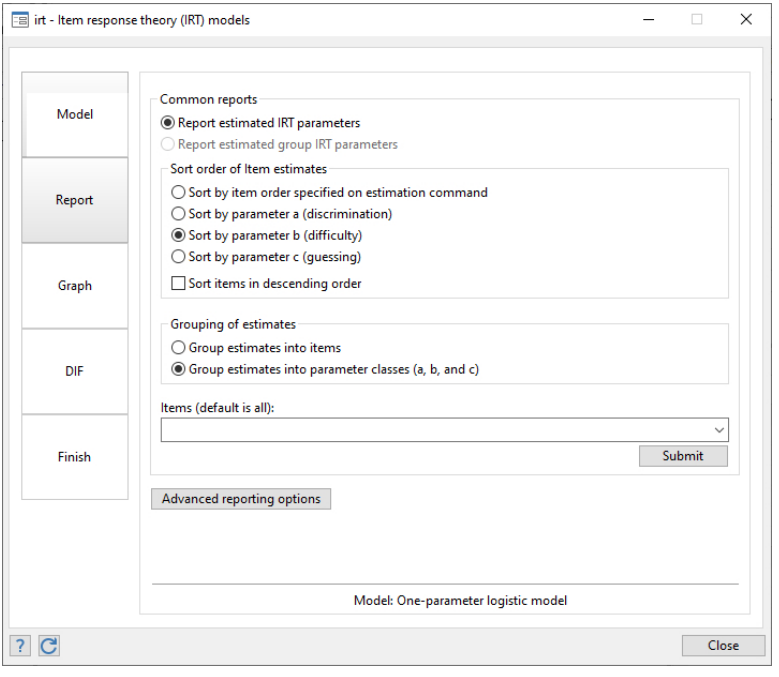


We click on **Fit model**, and the results appear in the Results window.

```
. irt 1pl q1 q2 q3 q4 q5 q6 q7 q8 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 = -4153.3609
Iteration 1:  Log likelihood =  -4142.374
Iteration 2:  Log likelihood = -4142.3516
Iteration 3:  Log likelihood = -4142.3516
One-parameter logistic model                    Number of obs = 800
Log likelihood = -4142.3516
```

		Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim		.852123	.0458445	18.59	0.000	.7622695	.9419765
q1	Diff	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2	Diff	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q3	Diff	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q4	Diff	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q5	Diff	1.652719	.1329494	12.43	0.000	1.392144	1.913295
q6	Diff	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7	Diff	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q8	Diff	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q9	Diff	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448

Next, we want to report the results sorted by difficulty. On the left of the Control Panel, we select the *Report* tab.

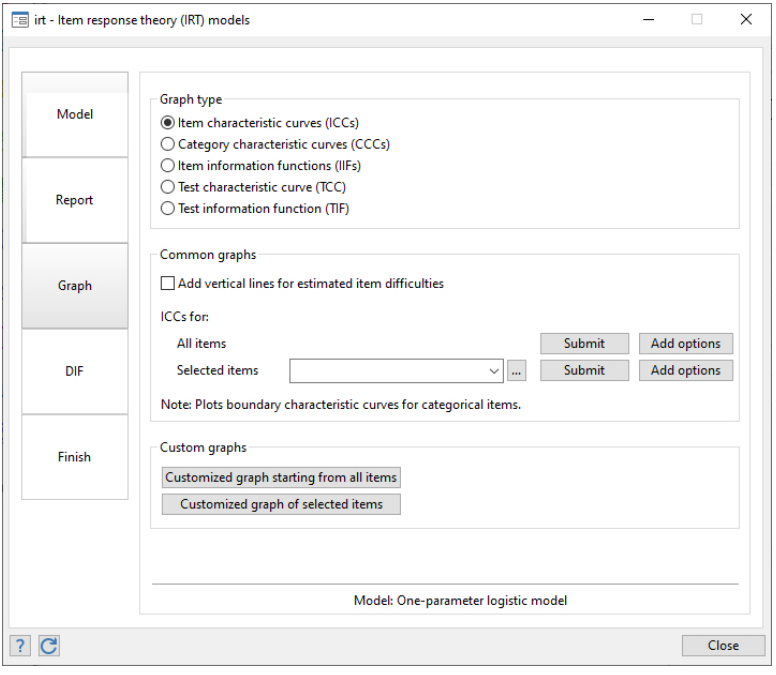


At the top, we select the *Report estimated IRT parameters* radio button. Then under *Sort order of item estimates*, we select the *Sort by parameter b (difficulty)* radio button, and under *Grouping of estimates*, we select the *Group estimates into parameter classes (a, b, and c)* radio button. By default, the report is created for all items in the model, so we do not need to select items `q1` through `q9` in the *Items* control. We click on the **Submit** button, and the new report appears in the Results window.

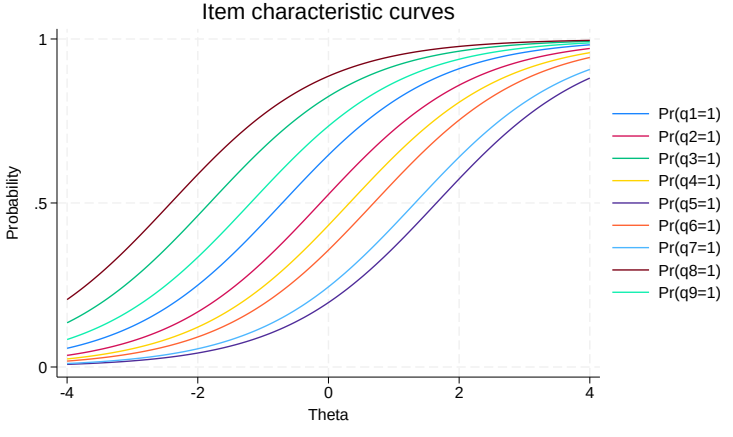
```
. estat report, sort(b) byparm
One-parameter logistic model                          Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
Diff						
q8	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q3	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q9	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448
q1	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q4	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q6	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q5	1.652719	.1329494	12.43	0.000	1.392144	1.913295

We are now ready to graph the ICCs for our items. On the left of the Control Panel, we select the *Graph* tab. Under *Graph type*, we select the *Item characteristic curves (ICCs)* radio button.

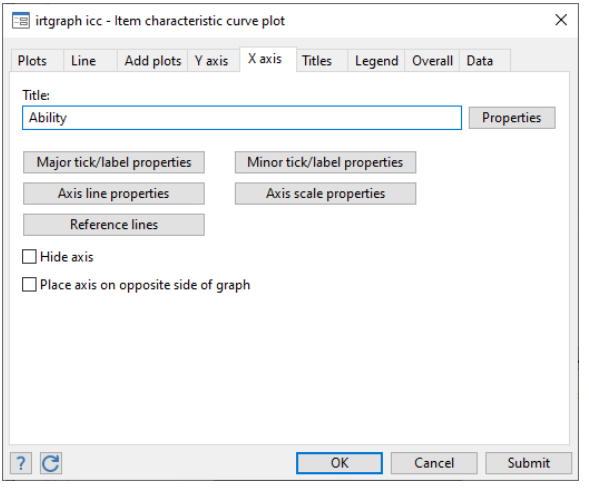


Then, in the *Common graphs* section, we simply click on the **Submit** button on the line requesting ICCs for all items to create the following graph.

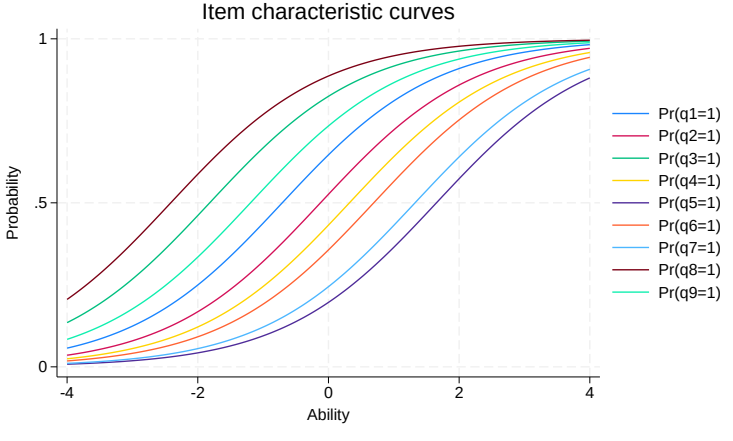


In [IRT] **irt**, lines were included in the graph to show the estimated difficulty for each item. If we want to add these lines, we can check *Add vertical lines for estimated item difficulties* before clicking on the **Submit** button.

The **Add options** button allows us to modify the appearance of the graph. We can change the title of the graph and its size, add a caption, change the title displayed along each axis, change the placement and appearance of labels on each axis, change the color and pattern of lines, change the placement and appearance of the legend, and much more. To demonstrate this, we change the title on the x axis from the default of θ to `Ability`. We click on the **Add options** button on the line for ICCs for all items. In the resulting dialog box, we select the *X axis* tab and type `Ability` in the *Title:* box.



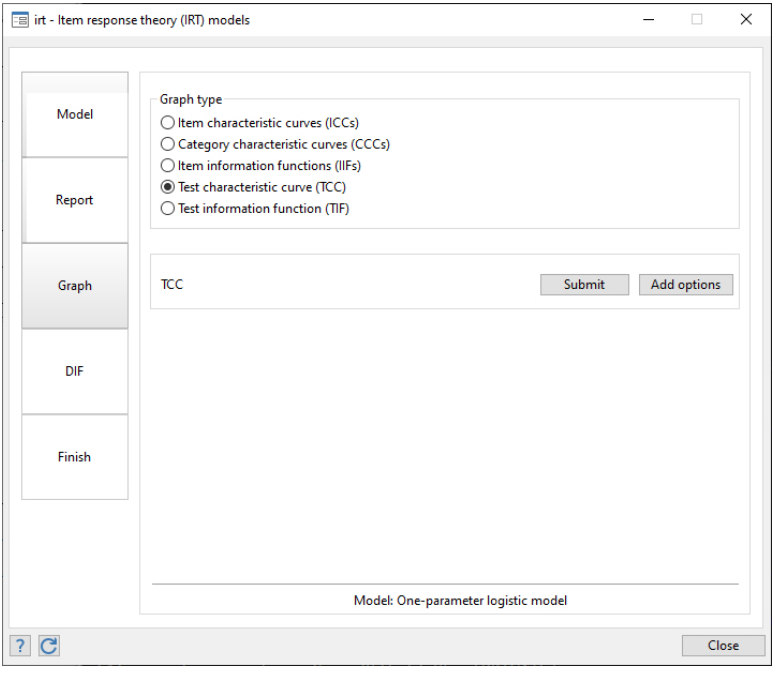
We click on **Submit** to re-create our previous graph but with the modified title on the x axis.



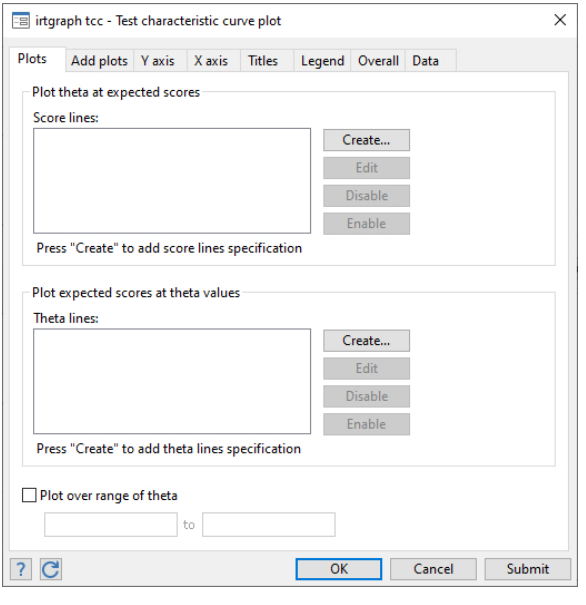
By clicking on **Submit** rather than on **OK**, we keep the dialog box open so that we can continue modifying the look of the graph. We will not make any further modifications at this point, so we will close the dialog box.

Back on the *Graph* tab of the Control Panel, there are two additional buttons in the *Custom graphs* section. Even further customization of graphs is available by clicking on the **Customized graph starting from all items** button or the **Customized graph of selected items** button. These allow us to change the appearance of each curve individually. For instance, we can specify a color or line pattern for one item or for a group of items. We can also make any of the modifications that are available through the **Add options** buttons.

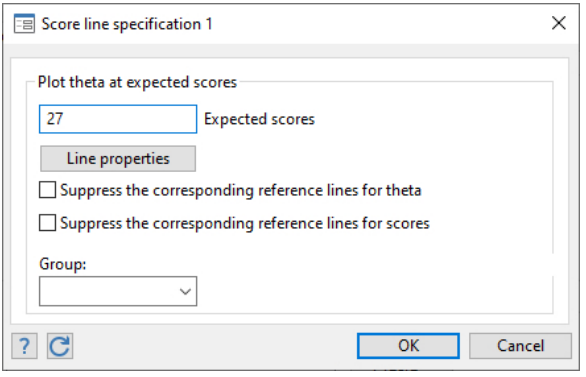
The example in [IRT] **irt** also graphs the TCCs. To create this graph, we select the *Test characteristic curve (TCC)* radio button from the *Graph* tab and click on the **Submit** button next to *TCC*.



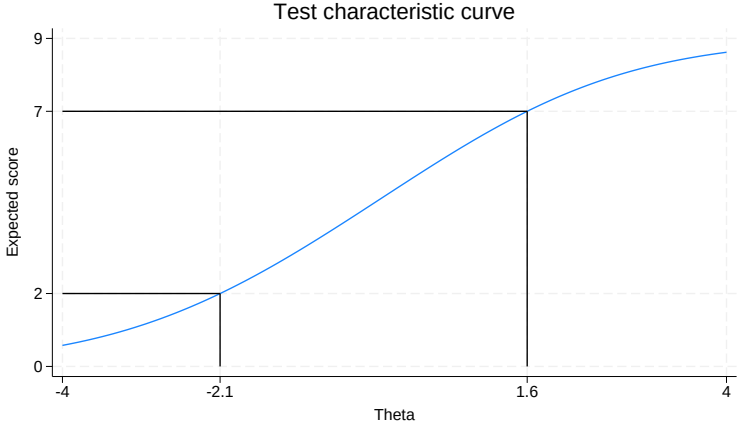
To add lines corresponding to the value of θ for expected scores of 2 and 7, we click on the **Add options** button. In the resulting dialog, we click on the **Create...** button next to the *Score lines:* box.



In the dialog box that opens, we type 2 7 in the box for *Expected scores*.



Then, we click on **OK** twice, and the following TCC graph is produced.



Although the example continues in [IRT] **irt**, we will stop at this point. We have now demonstrated the use of the *Model*, *Report*, and *Graph* tabs on the IRT Control Panel. You can use the Control Panel in a similar manner to fit any IRT model, to produce other types of reports, and to create graphs of ICCs, CCCs, TCCs, IIFs, and TIFs.

For reproducible research, you can create a command log containing all the commands that are issued from the Control Panel. To create the command log, click on the **Start command log** button on the *Model* tab before fitting a model. In the dialog box that opens, specify the name of the command log, say, `Ex_1pl`, to create a file named `Ex_1pl.txt` in your current working directory. Next, click on **OK**. Now you are ready to perform your analysis, storing all the commands that the Control Panel issues. Once you have completed your analysis, go to the *Finish* tab and click on the **Close command log** button. The `Ex_1pl.txt` file will contain all of your commands, which can be rerun to easily reproduce your analysis.

Reference

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Also see

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

irt 1pl — One-parameter logistic model

Description

`irt 1pl` fits one-parameter logistic (1PL) models to binary items. In the 1PL model, items vary in their difficulty but share the same discrimination parameter.

Quick start

1PL model for binary items `b1` to `b10`

```
irt 1pl 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 1pl 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

Remarks are presented under the following headings:

Overview

The following discussion is about how to use `irt` to fit 1PL models to binary items. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

In the 1PL model, item responses are typically of the form yes or no, correct or incorrect, agree or disagree, etc. Items are assumed to be equally discriminating and vary in their difficulty only. The probability of person j providing a positive answer to item i is given by

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

where a represents the discrimination common to all items, b_i represents the difficulty of item i, and θ_j is the latent trait of person j.

A related model attributable to Rasch (1960) uses a different parameterization of (1) with $a = 1$ and $\theta_j \sim N(0, \sigma^2)$. Although philosophically different from the 1PL model, the Rasch model produces equivalent predictions of the latent trait; see [SEM] **Example 28g** for a model fit using the Rasch parameterization. See *Item response theory* in [BAYES] **bayesmh** and Balov (2016) for examples demonstrating Bayesian estimation of this model.

▷ Example 1: Fitting a 1PL model

To illustrate the 1PL 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/r19/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 1PL model to binary items q1–q9 as follows:

```
. irt 1pl 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 = -4153.3609
Iteration 1:  Log likelihood =  -4142.374
Iteration 2:  Log likelihood = -4142.3516
Iteration 3:  Log likelihood = -4142.3516
One-parameter logistic model                    Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
q1						
Diff	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2						
Diff	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q3						
Diff	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q4						
Diff	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q5						
Diff	1.652719	.1329494	12.43	0.000	1.392144	1.913295
q6						
Diff	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7						
Diff	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q8						
Diff	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q9						
Diff	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448

Because the discrimination parameter is the same for all items, it is listed only once. The estimate of 0.85 suggests poor discrimination; that is, in the vicinity of a given difficulty estimate, any two students with distinct abilities would have similar predicted probabilities of success giving a correct answer to an item. If the items were highly discriminating, the calculated probabilities would be farther away from each other.

The estimates of the difficulty parameter correspond to the point on the ability 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.

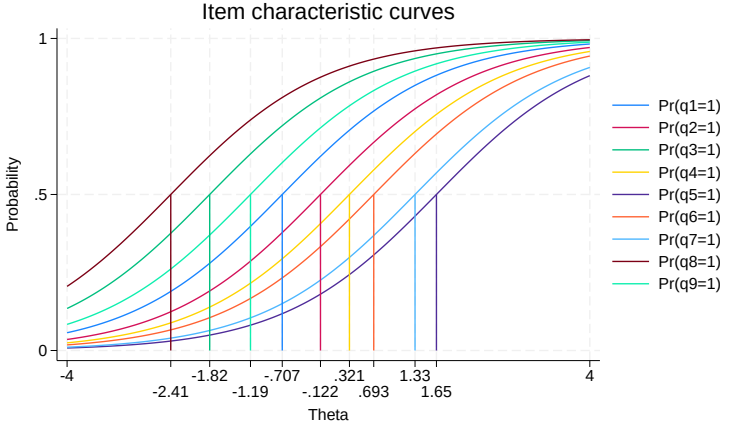
In the following, we use `estat report` to replay the table of estimated IRT parameters and control how the output is reported. We include the `byparm` option, which arranges the output by parameter rather than by item, and the `sort(b)` option, which displays the items in an ascending order of difficulty. This makes it easy to see that item `q8` is least difficult and item `q5` is most difficult.

```
. estat report, byparm sort(b)
One-parameter logistic model                    Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
Diff						
q8	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q3	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q9	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448
q1	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q4	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q6	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q5	1.652719	.1329494	12.43	0.000	1.392144	1.913295

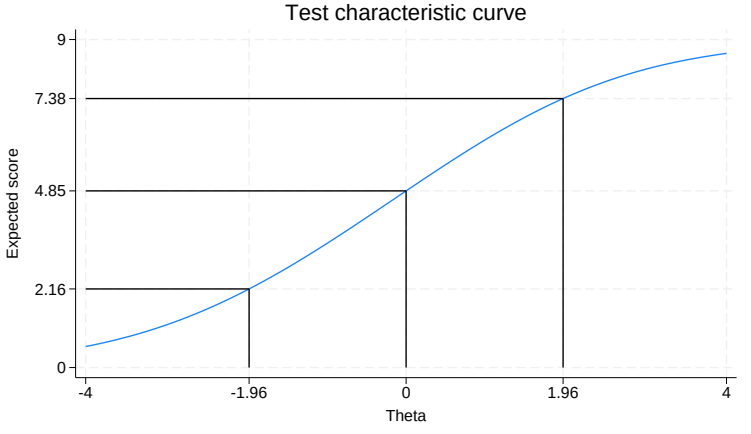
For the 1PL model, the ICC plots the probability of a successful response as a function of θ, using the estimated 1PL parameters. In the following, we use `irtgraph icc` to plot the ICCs. The `blocation` option adds a vertical line at the estimated difficulties; see [IRT] **irtgraph icc**. The `xlabel()` option creates more legible axis labels (see [G-3] ***axis_label_options***).

```
. irtgraph icc, blocation xlabel(, alt)
```



The TCC plots the expected score as a function of θ, using the estimated 1PL parameters. We use `irtgraph tcc` to plot the 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 of θ.

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



This plot tells us what kind of scores to expect from individuals with different levels of the latent trait. For example, we can expect above-average individuals to score 4.85 or above. Actually, no one is expected to score 4.85 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 2.16 and 7.38. Again, a more realistic statement is that we expect about 95% of randomly selected people to score between 2 and 7.

◁

Video example

Item response theory using Stata: One-parameter logistic (1PL) models

Stored results

`irt 1pl` 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)	1pl
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

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

where a represents discrimination, and b_i represents the difficulty of item i. `irt 1pl` fits the model using the slope-intercept form, so the probability for providing a correct answer is parameterized as

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

The transformation between these two parameterizations is

$$a = \alpha \qquad b_i = -\frac{\beta_i}{\alpha}$$

Let $p_{ij} = \Pr(Y_{ij} = 1|\alpha, \beta_i, \theta_j)$ 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 $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha, \beta_1, \ldots, \beta_I)$, 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*. https://blog.stata.com/2016/01/18/bayesian-binary-item-response-theory-models-using-bayesmh/.

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Rasch, G. 1960. *Probabilistic Models for Some Intelligence and Attainment Tests*. Copenhagen: Danish Institute of Educational Research.

Also see

[IRT] **irt 1pl postestimation** — Postestimation tools for irt 1pl

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

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

[SEM] Example 28g — One-parameter logistic IRT (Rasch) model

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

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

[U] 20 Estimation and postestimation commands

irt 1pl postestimation — Postestimation tools for irt 1pl

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt 1pl`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
`outcome(`*item*`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item*) specifies that predictions for *item* be calculated. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(`#`)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(`#`)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(`#`)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt 1pl**.

When the `marginal` option is specified, the predicted probability is computed as

$$\hat{p}_{ij} = \int_{-\infty}^{\infty} \frac{\exp(\hat{\alpha}\,\theta_j + \hat{\beta}_i)}{1 + \exp(\hat{\alpha}\,\theta_j + \hat{\beta}_i)} \phi(\theta_j)\, d\theta_j$$

where $\hat{\alpha}$ and $\hat{\beta}_i$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\alpha}\,\tilde{\theta}_j + \hat{\beta}_i$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\beta}_i$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ij} = \frac{\exp(\hat{z}_{ij})}{1 + \exp(\hat{z}_{ij})}$$

Also see

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

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

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

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/r19/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` $= 1.62$).

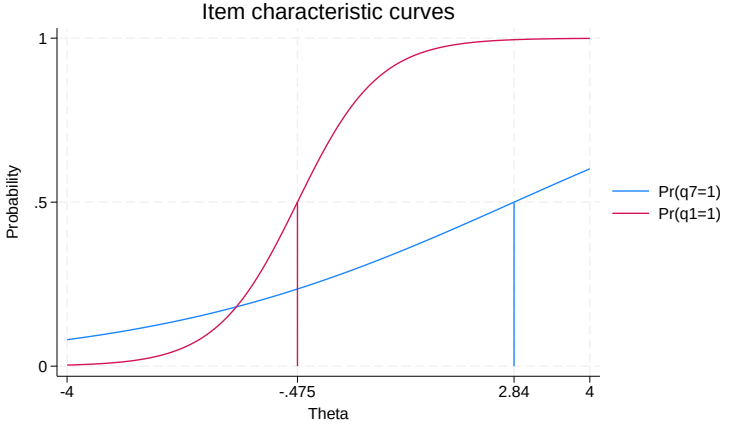
```
. estat report, byparm sort(a)
Two-parameter logistic model                    Number of obs = 800
Log likelihood = -4118.4697
```

	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.

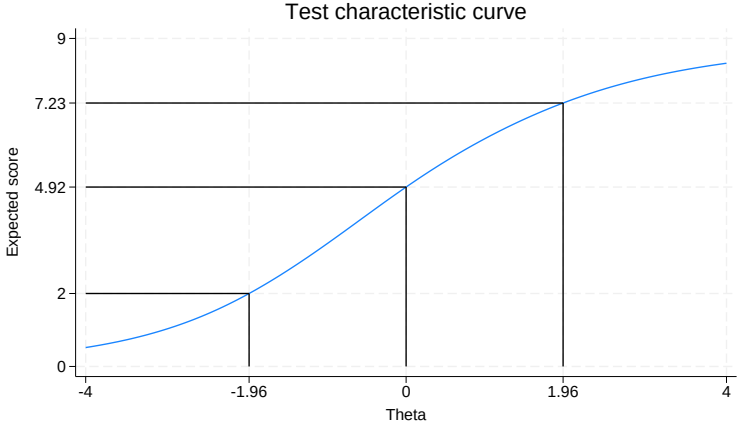
```
. irtgraph icc q7 q1, blocation
```



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*. https://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*, edited by F. M. Lord and M. R. Novick, 395–479. Reading, MA: Addison–Wesley.

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

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**

irt 2pl postestimation — Postestimation tools for irt 2pl

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt 2pl`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
`outcome(`*item*`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item*) specifies that predictions for *item* be calculated. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(#)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(#)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(#)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt 2pl**.

When the `marginal` option is specified, the predicted probability is computed as

$$\hat{p}_{ij} = \int_{-\infty}^{\infty} \frac{\exp(\hat{\alpha}_i\,\theta_j + \hat{\beta}_i)}{1 + \exp(\hat{\alpha}_i\,\theta_j + \hat{\beta}_i)}\,\phi(\theta_j)\,d\theta_j$$

where $\hat{\alpha}_i$ and $\hat{\beta}_i$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\alpha}_i\,\tilde{\theta}_j + \hat{\beta}_i$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\beta}_i$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ij} = \frac{\exp(\hat{z}_{ij})}{1 + \exp(\hat{z}_{ij})}$$

Also see

[IRT] **irt 2pl** — Two-parameter logistic model

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt 3pl — Three-parameter logistic model

Description

`irt 3pl` fits three-parameter logistic (3PL) models to binary items. In the 3PL model, items vary in their difficulty and discrimination and the possibility of guessing is allowed.

Quick start

3PL model for binary items `b1` to `b10`

```
irt 3pl 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 3pl 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
`sepguessing`	estimate a separate pseudoguessing parameter for each item
`gsepguessing`	estimate separate pseudoguessing parameters for each group
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.

`sepguessing` specifies that a separate pseudoguessing parameter be estimated for each item. This is a seldom used option; see the technical note below.

`gsepguessing` specifies that separate pseudoguessing parameters be estimated for each group. This option is allowed only when fitting a group model.

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

Remarks are presented under the following headings:

Overview

The following discussion is about how to use `irt` to fit (3PL) models to binary items. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

In the 3PL 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, and the model accommodates the possibility of guessing on a test. The probability of person j providing a positive answer to item i is given by

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

where a_i represents the discrimination of item i, b_i represents the difficulty of item i, c_i represents the pseudoguessing parameter, and θ_j is the latent trait of person j. By default, the c_i are constrained to be the same across all items; see the technical note below.

Although (1) is not in logistic form, the model is commonly referred to as a three-parameter logistic model.

The 3PL model was proposed by Birnbaum (1968). An earlier three-parameter model with a probit link was developed by Finney (1952).

❑ Technical note

By default, `irt 3pl` constrains the pseudoguessing parameter to be the same across all items. You can use the advanced option `sepguessing` to request a separate pseudoguessing parameter for each item. We do not recommend this option because this version of the 3PL model is plagued with identification problems; see, for example, Samejima (1973), Holland (1990), Yen, Burket, and Sykes (1991), Maris (2002), and San Martín, Rolin, and Castro (2013).

The `sepguessing` option can be useful in the context of hybrid IRT models, where separate pseudoguessing parameters can be estimated for a subset of items; see example 2 in [IRT] **irt hybrid**.

See Balov (2016) for an example of Bayesian estimation of a 3PL model with separate pseudoguessing parameters.

❑

▷ Example 1: Fitting a 3PL model

To illustrate the 3PL 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/r19/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 3PL model to binary items q1–q9 as follows:

```
. irt 3pl q1-q9
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -5322.8824
Iteration 1:  Log likelihood = -4317.9868
Iteration 2:  Log likelihood = -4273.6659
Iteration 3:  Log likelihood = -4269.7862
Iteration 4:  Log likelihood = -4269.7825
Iteration 5:  Log likelihood = -4269.7825
Fitting full model:
Iteration 0:  Log likelihood = -4226.5553  (not concave)
Iteration 1:  Log likelihood = -4126.9014  (not concave)
Iteration 2:  Log likelihood = -4120.7233
Iteration 3:  Log likelihood = -4117.2619
Iteration 4:  Log likelihood = -4116.3931
Iteration 5:  Log likelihood = -4116.3434
Iteration 6:  Log likelihood = -4116.3404
Iteration 7:  Log likelihood = -4116.3404
```

```
Three-parameter logistic model                          Number of obs = 800
Log likelihood = -4116.3404
```

		Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1							
	Discrim	1.911892	.3633509	5.26	0.000	1.199737	2.624047
	Diff	-.3040607	.0970816	-3.13	0.002	-.4943372	-.1137842
q2							
	Discrim	.750889	.1414085	5.31	0.000	.4737334	1.028045
	Diff	.1506376	.1667842	0.90	0.366	-.1762535	.4775287
q3							
	Discrim	.9674965	.1682051	5.75	0.000	.6378205	1.297172
	Diff	-1.508912	.2358768	-6.40	0.000	-1.971222	-1.046602
q4							
	Discrim	.9846883	.1860968	5.29	0.000	.6199454	1.349431
	Diff	.5726226	.1491574	3.84	0.000	.2802794	.8649659
q5							
	Discrim	1.439631	.4426227	3.25	0.001	.5721063	2.307156
	Diff	1.605677	.2144335	7.49	0.000	1.185395	2.025959
q6							
	Discrim	1.369119	.3249596	4.21	0.000	.7322101	2.006028
	Diff	.7818615	.1236333	6.32	0.000	.5395447	1.024178
q7							
	Discrim	.4823135	.1727569	2.79	0.005	.1437162	.8209108
	Diff	3.010922	.8924986	3.37	0.001	1.261656	4.760187
q8							
	Discrim	1.436069	.2482751	5.78	0.000	.9494586	1.922679
	Diff	-1.594747	.1918747	-8.31	0.000	-1.970815	-1.21868
q9							
	Discrim	.6772551	.1314525	5.15	0.000	.419613	.9348971
	Diff	-1.213933	.2661804	-4.56	0.000	-1.735637	-.6922291
	Guess	.0904473	.0359669			.0199534	.1609412

In the 3PL model, each test item has its own parameter estimates for discrimination and difficulty. The estimated common pseudoguessing parameter is reported at the end of the table.

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(b)` option, which displays the items in an ascending order of difficulty. This makes it easy to see that item `q8` is least difficult and item `q7` is most difficult.

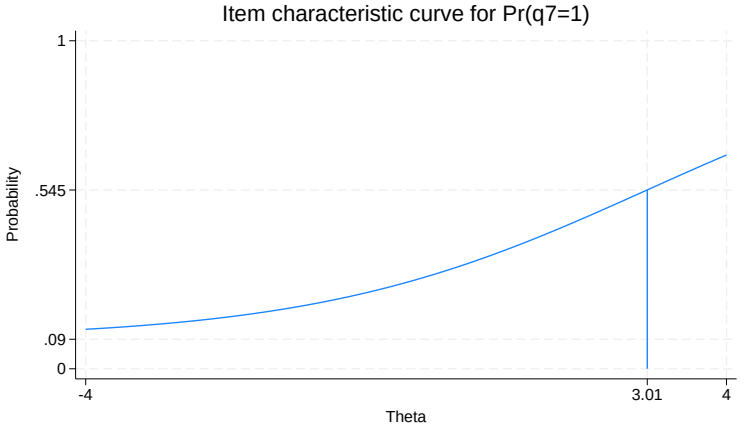
```
. estat report, byparm sort(b)
Three-parameter logistic model                  Number of obs = 800
Log likelihood = -4116.3404
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim						
q8	1.436069	.2482751	5.78	0.000	.9494586	1.922679
q3	.9674965	.1682051	5.75	0.000	.6378205	1.297172
q9	.6772551	.1314525	5.15	0.000	.419613	.9348971
q1	1.911892	.3633509	5.26	0.000	1.199737	2.624047
q2	.750889	.1414085	5.31	0.000	.4737334	1.028045
q4	.9846883	.1860968	5.29	0.000	.6199454	1.349431
q6	1.369119	.3249596	4.21	0.000	.7322101	2.006028
q5	1.439631	.4426227	3.25	0.001	.5721063	2.307156
q7	.4823135	.1727569	2.79	0.005	.1437162	.8209108
Diff						
q8	-1.594747	.1918747	-8.31	0.000	-1.970815	-1.21868
q3	-1.508912	.2358768	-6.40	0.000	-1.971222	-1.046602
q9	-1.213933	.2661804	-4.56	0.000	-1.735637	-.6922291
q1	-.3040607	.0970816	-3.13	0.002	-.4943372	-.1137842
q2	.1506376	.1667842	0.90	0.366	-.1762535	.4775287
q4	.5726226	.1491574	3.84	0.000	.2802794	.8649659
q6	.7818615	.1236333	6.32	0.000	.5395447	1.024178
q5	1.605677	.2144335	7.49	0.000	1.185395	2.025959
q7	3.010922	.8924986	3.37	0.001	1.261656	4.760187
Guess	.0904473	.0359669			.0199534	.1609412

The estimate of the pseudoguessing parameter is 0.09, which suggests a modest degree of guessing on the test. The pseudoguessing parameter represents the smallest probability of a correct response. Thus, according to this model, even the least able student has, at minimum, a 9% chance of responding correctly on any given item.

After `irt 3pl`, we can use `irtgraph icc` to plot the ICCs using the estimated 3PL parameters; see [IRT] **irtgraph icc**. To focus on the most difficult item, as reported by `estat report`, we restrict the plot to item `q7`. We use option `blocation` to add a vertical line at the estimated difficulty and option `ylabel()` to change the default labeling of the y axis to include the lower asymptote and the midpoint probability, where θ equals the estimated difficulty for `q7`.

```
. irtgraph icc q7, blocation ylabel(0 0.09 0.545 1)
```



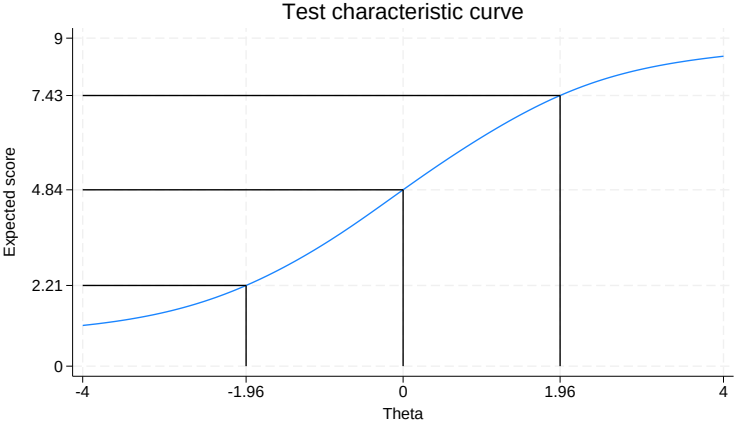
Notice that the estimate of the pseudoguessing parameter is now a lower asymptote for the plotted ICC. Also, because of the pseudoguessing parameter, the midpoint probability, where θ equals the estimated difficulty for `q7`, is

$$\hat{c} + (1 - \hat{c}) \times \frac{1}{2} = 0.09 + 0.91 \times \frac{1}{2} = 0.545$$

instead of 0.5, as in the case of 1PL and 2PL models.

The TCC plots the expected score as a function of θ, using the estimated 3PL parameters. We use `irtgraph tcc` to plot the TCC. For 9 binary items, it is clear that the total score ranges from 0 to 9; however, because of the pseudoguessing parameter, the minimum expected score is $\hat{c} \times 9 = 0.09 \times 9 = 0.81$. 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 the latent trait. For example, we can expect above-average individuals to score 4.84 or above. Actually, no one is expected to score 4.84 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 2.21 and 7.43. 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: Three-parameter logistic (3PL) models

Stored results

`irt 3pl` 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(sepguess1)`	1 if model contains a separate pseudoguessing parameter
`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)	3pl
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 (the latent trait) providing a correct response to item i is given by

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

where a_i represents the discrimination of item i, b_i represents the difficulty of item i, and c_i represents the pseudoguessing parameter. `irt 3pl` fits the model using the slope-intercept form, so the probability for providing a correct answer is parameterized as

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

The transformation between these two parameterizations is

$$a_i = \alpha_i \qquad b_i = -\frac{\beta_i}{\alpha_i} \qquad c_i = \frac{\exp(\gamma_i)}{1 + \exp(\gamma_i)}$$

By default, the γ_i (and thus the c_i) are constrained to be the same across all items.

Let $p_{ij} \equiv \Pr(Y_{ij} = 1|\alpha_i, \beta_i, \gamma_i, \theta_j)$ 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 $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha_1, \ldots, \alpha_I, \beta_1, \ldots, \beta_I, \gamma_1, \ldots, \gamma_I)$, 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*. https://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*, edited by F. M. Lord and M. R. Novick, 395–479. Reading, MA: Addison–Wesley.

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Finney, D. J. 1952. *Probit Analysis: A Statistical Treatment of the Sigmoid Response Curve*. 2nd ed. New York: Cambridge University Press.

Holland, P. W. 1990. The Dutch identity: A new tool for the study of item response models. *Psychometrika* 55: 5–18. https://doi.org/10.1007/BF02294739.

Maris, G. 2002. Concerning the identification of the 3PL model. Tech. Rep. 2002-3, CITO National Institute for Educational Measurement, Arnhem, The Netherlands.

Samejima, F. 1973. A comment on Birnbaum's three-parameter logistic model in the latent trait theory. *Psychometrika* 38: 221–233. https://doi.org/10.1007/BF02291115.

San Martín, E., J.-M. Rolin, and L. M. Castro. 2013. Identification of the 1PL model with guessing parameter: Parametric and semi-parametric results. *Psychometrika* 78: 341–379. https://doi.org/10.1007/s11336-013-9322-8.

Yen, W. M., G. R. Burket, and R. C. Sykes. 1991. Nonunique solutions to the likelihood equation for the three-parameter logistic model. *Psychometrika* 56: 39–54. https://doi.org/10.1007/BF02294584.

Also see

[IRT] **irt 3pl postestimation** — Postestimation tools for irt 3pl

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

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

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

[U] 20 Estimation and postestimation commands

irt 3pl postestimation — Postestimation tools for irt 3pl

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt 3pl`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
`outcome(`*item*`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item*) specifies that predictions for *item* be calculated. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(#)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(#)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(#)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt 3pl**.

When the `marginal` option is specified, the predicted probability is computed as

$$\hat{p}_{ij} = \hat{c}_i + (1-\hat{c}_i)\int_{-\infty}^{\infty} \frac{\exp(\hat{\alpha}_i\,\theta_j + \hat{\beta}_i)}{1+\exp(\hat{\alpha}_i\,\theta_j + \hat{\beta}_i)}\phi(\theta_j)\,d\theta_j$$

where $\hat{\alpha}_i$, $\hat{\beta}_i$, and $\hat{\gamma}_i$ are the estimated parameters in the slope-intercept parameterization, and

$$\hat{c}_i = \frac{\exp(\hat{\gamma}_i)}{1+\exp(\hat{\gamma}_i)}$$

The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\alpha}_i\,\tilde{\theta}_j + \hat{\beta}_i$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\beta}_i$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ij} = \hat{c}_i + (1-\hat{c}_i)\frac{\exp(\hat{z}_{ij})}{1+\exp(\hat{z}_{ij})}$$

Also see

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

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt grm — Graded response model

Description

`irt grm` fits graded response models (GRMs) to ordinal items. In the GRM, items vary in their difficulty and discrimination. This model is an extension of the 2PL model to ordered categorical items.

Quick start

GRM for ordinal items `o1` to `o5`

```
irt grm o1-o5
```

Report items sorted by discrimination

```
estat report, sort(a)
```

Plot CCCs for `o1`

```
irtgraph icc o1
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irt grm 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

Remarks are presented under the following headings:

Overview
Video example

Overview

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

In the GRM, item responses are categorical and ordered, for example, “poor”, “good”, and “excellent” or “strongly disagree”, “disagree”, “agree”, and “strongly agree”. If there are only two outcomes, the GRM is equivalent to the 2PL model; see [IRT] **irt 2pl**. If the item responses are not ordered, see [IRT] **irt nrm**.

The GRM allows the ordered categories to vary between items; however, to keep the following discussion from being overly complicated, we will assume the outcome levels for all items are given by $k = 0, 1, \ldots, K$.

In the GRM, each item is modeled with its own discrimination parameter and cutpoints that identify boundaries between the ordered outcomes. The probability of observing outcome k or higher for item i and person j is given by

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

where a_i represents the discrimination of item i, b_{ik} is the kth cutpoint for item i, and θ_j is the latent trait of person j. The cutpoint b_{ik} can be considered the difficulty of responding with category k or higher for item i.

The GRM is defined in terms of cumulative probabilities, but we can calculate the probability of observing outcome k as

$$\Pr(Y_{ij} = k|\theta_j) = \Pr(Y_{ij} \geq k|\theta_j) - \Pr(Y_{ij} \geq k+1|\theta_j)$$

where we take $\Pr(Y_{ij} \geq 0) = 1$ and $\Pr(Y_{ij} > K) = 0$. Because of the additional calculation step required to obtain the probability of observing a particular outcome, the GRM is an indirect IRT model, also known as a difference model; see Thissen and Steinberg (1986).

The GRM was proposed by Samejima (1969). In the multilevel literature, the GRM is known as the cumulative logit model; see [ME] **meologit**. When no latent variable is present, the model for a single item is known as the proportional odds model; see [R] **ologit**.

▷ Example 1: Fitting a GRM

To illustrate the GRM, we use the data from Zheng and Rabe-Hesketh (2007). `charity.dta` contains five survey questions, `ta1` through `ta5`, measuring faith and trust in charity organizations. Responses are strongly agree (0), agree (1), disagree (2), and strongly disagree (3). Higher scores indicate higher levels of distrust. Here we list the first five observations.

```
. use https://www.stata-press.com/data/r19/charity
(Data from Zheng & Rabe-Hesketh (2007))
. list in 1/5, nolabel
```

	ta1	ta2	ta3	ta4	ta5
1.	.	2	1	1	.
2.	0	0	0	0	0
3.	1	1	2	0	2
4.	1	2	2	0	1
5.	.	1	1	1	1

Looking across the first row, we see that the first respondent did not provide an answer to items `ta1` and `ta5`, answered 2 on item `ta2`, and answered 1 on items `ta3` and `ta4`. All `irt` commands exclude missing items for a given observation from the likelihood calculation but keep the nonmissing items for that observation. If you wish to remove the entire observation from the model, add the `listwise` option at estimation time.

We fit a GRM as follows:

```
. irt grm ta1-ta5
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -5559.6414
Iteration 1:  Log likelihood = -5473.9434
Iteration 2:  Log likelihood = -5467.4082
Iteration 3:  Log likelihood = -5467.3926
Iteration 4:  Log likelihood = -5467.3926
Fitting full model:
Iteration 0:  Log likelihood = -5271.0634
Iteration 1:  Log likelihood = -5162.5917
Iteration 2:  Log likelihood = -5159.2947
Iteration 3:  Log likelihood = -5159.2791
Iteration 4:  Log likelihood = -5159.2791
Graded response model                                   Number of obs = 945
Log likelihood = -5159.2791
```

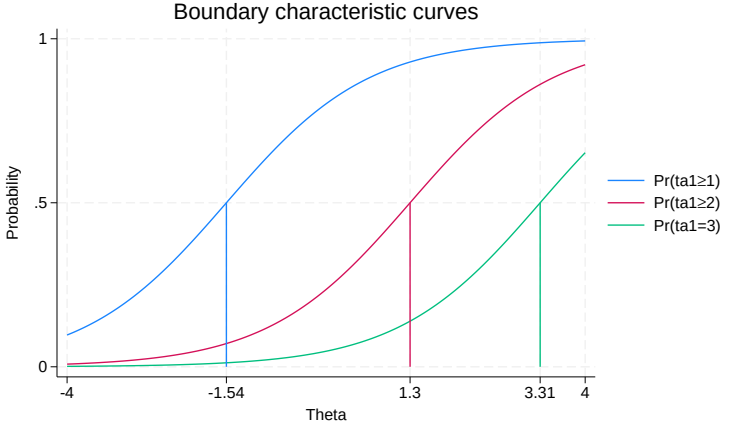
	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
ta1						
Discrim	.907542	.0955772	9.50	0.000	.7202142	1.09487
Diff						
>=1	-1.540098	.1639425			-1.861419	-1.218776
>=2	1.296135	.1427535			1.016343	1.575927
=3	3.305059	.3248468			2.668371	3.941747
ta2						
Discrim	.9434675	.0967483	9.75	0.000	.7538444	1.133091
Diff						
>=1	-1.661331	.167878			-1.990366	-1.332296
>=2	.0068314	.082222			-.1543208	.1679836
=3	2.531091	.2412513			2.058247	3.003935
ta3						
Discrim	1.734201	.1554383	11.16	0.000	1.429548	2.038855
Diff						
>=1	-1.080079	.0835119			-1.243759	-.9163983
>=2	1.016567	.0796635			.8604297	1.172705
=3	2.232606	.1497814			1.93904	2.526172
ta4						
Discrim	1.93344	.1857629	10.41	0.000	1.569351	2.297528
Diff						
>=1	-.3445057	.0578468			-.4578833	-.2311282
>=2	1.466254	.0983823			1.273428	1.65908
=3	2.418954	.162392			2.100672	2.737237
ta5						
Discrim	1.42753	.1263962	11.29	0.000	1.179798	1.675262
Diff						
>=1	-.8552358	.0833158			-1.018532	-.6919399
>=2	.6805315	.07469			.5341418	.8269211
=3	2.074243	.1538858			1.772632	2.375853

Because the GRM is basically an ordered logistic model, each item's difficulty parameters are naturally estimated in an increasing order. The difficulties represent a point at which a person with trait level $\theta_j = b_{ik}$ has a 50% chance of responding in category k or higher. We make cumulative comparisons because the model is defined in terms of cumulative probabilities.

For example, looking at the estimated parameters of item `ta1`, we see that a person with $\theta = -1.54$ has a 50% chance of answering 0 versus greater than or equal to 1, a person with $\theta = 1.30$ has a 50% chance of answering 0 or 1 versus greater than or equal to 2, and a person with $\theta = 3.31$ has a 50% chance of answering 0, 1, or 2 versus 3.

To illustrate this, we plot the BCCs as a function of θ for `ta1` using the estimated GRM parameters. The `blocation` option adds a vertical line at the estimated difficulties; see [IRT] **irtgraph icc**.

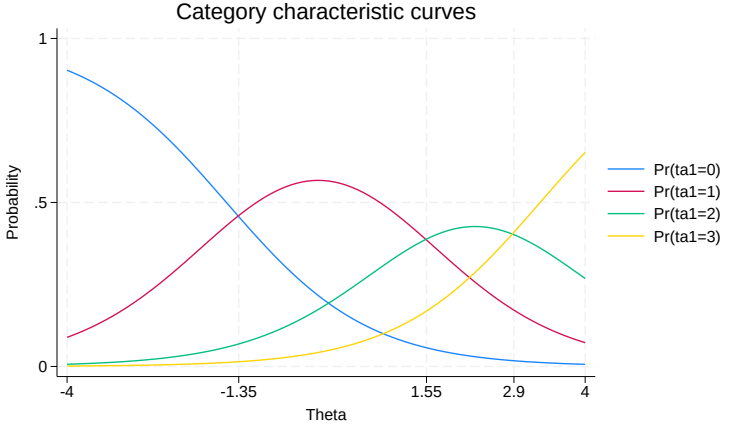
```
. irtgraph icc ta1, blocation
```



We see that the estimated difficulty parameters correspond to the point on the latent trait scale at which $\Pr(Y \geq k|\theta) = 0.5$. You can think of these curves as item characteristic curves where each curve dichotomizes the ordered responses into successive $\Pr(Y \geq k)$ and $\Pr(Y < k)$ categories. The estimated discrimination parameter for `ta1` is 0.91; thus, the curves have relatively flat slopes.

We can also plot category probabilities, $\Pr(Y = k)$, as a function of θ, which in fact is the default behavior of `irtgraph icc`. For categorical responses, such plots are called category characteristic curves (CCCs). Here we plot the CCCs for item `ta1`.

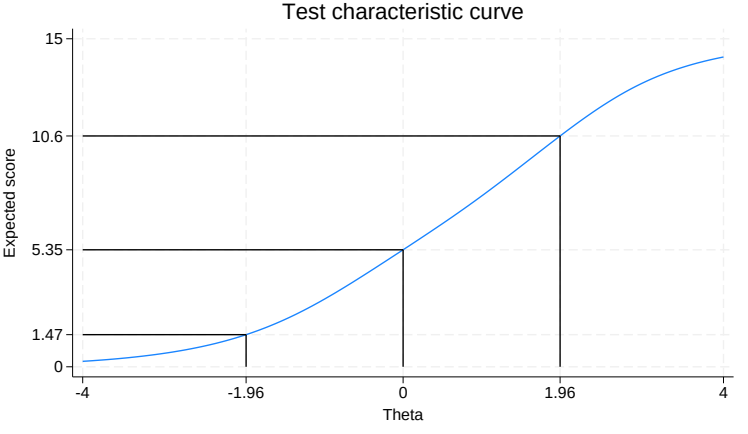
```
. irtgraph icc ta1, xlabel(-4 -1.35 1.55 2.9 4)
```



The graph shows that respondents with the latent trait level below approximately −1.35 are most likely to respond in the first category, respondents with the latent trait level between approximately −1.35 and 1.55 are most likely to respond in the second category, and so on.

We use `irtgraph tcc` to plot the TCC using the estimated GRM parameters; see [IRT] **irtgraph tcc**. Because we have 5 items, each coded 0 to 3, the total score ranges from 0 to 15. 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 the latent trait (trust in charities).

For example, we can expect above-average individuals to score 5.35 or above. Actually, no one is expected to score exactly 5.35 on this survey, so a more realistic statement is that we expect above-average individuals to score above 5 out of a possible score of 15.

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 1.47 and 10.6. Again, a more realistic statement is that we expect about 95% of randomly selected people to score from 2 to 10, which can be interpreted that most people either trust or slightly distrust charities.

◁

Video example

Item response theory using Stata: Graded response models (GRMs)

Stored results

irt grm 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(k_cat#)	number of categories for the #th item, ordinal
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)	grm
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(cat#)	categories for the #th item, ordinal
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. Without loss of generality, we will assume all items take on the ordered categories, $k = 0, 1, \ldots, K$.

Using the IRT parameterization, we see that the probability of person j with latent trait level θ_j (the latent trait) providing response k or above for item i is given by

$$\Pr(Y_{ij} \geq k|a_i, \mathbf{b}_i, \theta_j) = \frac{\exp\{a_i(\theta_j - b_{ik})\}}{1 + \exp\{a_i(\theta_j - b_{ik})\}}$$

where a_i represents the discrimination of item i, $\mathbf{b}_i = (b_{i1}, \ldots, b_{iK})$ represent the difficulties that distinguish the ordered categories of item i, and it is understood that $\Pr(Y_{ij} \geq 0|a_i, \mathbf{b}_i, \theta_j) = 1$ and $\Pr(Y_{ij} > K|a_i, \mathbf{b}_i, \theta_j) = 0$. The probability of observing outcome k is then

$$\Pr(Y_{ij} = k|a_i, \mathbf{b}_i, \theta_j) = \Pr(Y_{ij} \geq k|a_i, \mathbf{b}_i, \theta_j) - \Pr(Y_{ij} \geq k+1|a_i, \mathbf{b}_i, \theta_j)$$

irt grm fits the model using the slope-intercept form, so the probability for providing response k or above is parameterized as

$$\Pr(Y_{ij} \geq k|\alpha_i, \boldsymbol{\beta}_i, \theta_j) = \frac{\exp(\alpha_i\theta_j - \beta_{ik})}{1 + \exp(\alpha_i\theta_j - \beta_{ik})}$$

The transformation between these two parameterizations is

$$a_i = \alpha_i \qquad b_{ik} = \frac{\beta_{ik}}{\alpha_i}$$

Let y_{ij} be the observed response for Y_{ij} and $p_{ij} = \Pr(Y_{ij} = y_{ij}|\alpha_i, \boldsymbol{\beta}_i, \theta_j)$. Conditional on θ_j, the item responses are assumed to be independent, so the conditional density for person j is given by

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

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha_1, \ldots, \alpha_I, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_I)$, 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

Samejima, F. 1969. Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monograph Supplement*, no. 17.

Thissen, D., and L. Steinberg. 1986. A taxonomy of item response models. *Psychometrika* 51: 567–577. https://doi.org/10.1007/BF02295596.

Zheng, X., and S. Rabe-Hesketh. 2007. Estimating parameters of dichotomous and ordinal item response models with gllamm. *Stata Journal* 7: 313–333.

Also see

[IRT] **irt grm postestimation** — Postestimation tools for irt grm

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

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

[IRT] **irt rsm** — Rating scale model

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

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

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

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt grm`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* *#*`)` may also be specified as `outcome(`*#.item*`)` or `outcome(`*item* `#`*#*`)`. `outcome(`*item* `#3)` means the third outcome value. `outcome(`*item* `#3)` would mean the same as `outcome(`*item* `4)` if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(`#`)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(`#`)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(`#`)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt grm**.

When the `marginal` option is specified, the predicted probability for item i, person j, and outcome k is computed as

$$\hat{p}_{ijk} = \int_{-\infty}^{\infty} \Pr(Y_{ij} = k|\hat{\alpha}_i, \widehat{\boldsymbol{\beta}}_i, \theta_j)\phi(\theta_j)\, d\theta_j$$

where $\hat{\alpha}_i$ and $\widehat{\boldsymbol{\beta}}_i$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ij} = \hat{\alpha}_i\, \tilde{\theta}_j$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ij} = 0$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ijk} = \Pr(Y_{ij} = k|\hat{\alpha}_i, \widehat{\boldsymbol{\beta}}_i, \tilde{\theta}_j)$$

Also see

[IRT] **irt grm** — Graded response model

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt nrm — Nominal response model

Description

`irt nrm` fits nominal response models (NRMs) to categorical items. In the NRM, items vary in their difficulty and discrimination.

Quick start

NRM for nominal items `n1` to `n5`

```
irt nrm n1-n5
```

Plot CCCs for `n1`

```
irtgraph icc n1
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irt nrm 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

Remarks are presented under the following headings:

Overview

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

The NRM is used for nominally scored responses. The responses are allocated to mutually exclusive, exhaustive, and nonordered categories. For example, responses to a behavioral question may be recorded as "yes", "no", "maybe", and "refused to say", and the actual score has no meaning other than to designate the response category. If there are only two outcomes, the NRM is equivalent to the 2PL model; see [IRT] **irt 2pl**. If the item responses are ordered, see [IRT] **irt grm**, [IRT] **irt pcm**, or [IRT] **irt rsm**.

The NRM allows the categories to vary between items; however, to keep the following discussion from being overly complicated, we will assume the outcome levels for all items are given by $k = 1, \ldots, K$.

In the NRM, the items are modeled each with their own collection of discrimination and "difficulty" parameters using the multinomial logistic model. We quote "difficulty" here to acknowledge that in the context of the NRM as implemented in `irt nrm`, the difficulty parameter measures the propensity to choose a given item category instead of the base outcome. For item i, the probability of person j choosing category k on item i is

$$\Pr(Y_{ij} = k|\theta_j) = \frac{\exp\{a_{ik}(\theta_j - b_{ik})\}}{\sum_{h=1}^{K} \exp\{a_{ih}(\theta_j - b_{ih})\}} \qquad \theta_j \sim N(0,1)$$

where a_{ik} represents the discrimination of category k for item i, b_{ik} represents the difficulty of category k for item i, and θ_j is the latent trait of person j. `irt nrm` assigns the first outcome as the base outcome with which other parameters will be compared; this implies the constraint $a_{i1} = 0$ and $b_{i1} = 0$ for each item i. With this constraint, a_{ik} and b_{ik} are the discrimination and difficulty to choose category k relative to the first category.

The NRM was proposed by Bock (1972). The slope-intercept parameterization of the NRM was proposed by Baker and Kim (2004). When no latent variable is present, the model for a single item is known as the multinomial logistic model; see [R] **mlogit**.

▷ Example 1: Fitting an NRM

To illustrate the NRM, we use the data from de Ayala (2022). `science.dta` contains four multiple-choice questions from a physical science test, `q1` through `q4`, with each response assigned to an unordered category of 1, 2, 3, or 4.

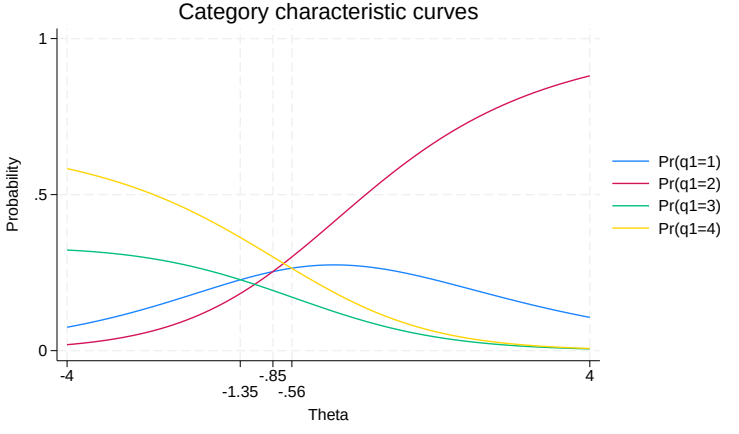
```
. use https://www.stata-press.com/data/r19/science
(Physical science data from de Ayala (2009))
. irt nrm q1-q4
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -9256.1514
Iteration 1:  Log likelihood = -9256.1514
Fitting full model:
Iteration 0:  Log likelihood = -9287.6878  (not concave)
Iteration 1:  Log likelihood = -9221.5198
Iteration 2:  Log likelihood = -9207.5015  (not concave)
Iteration 3:  Log likelihood = -9165.7851
Iteration 4:  Log likelihood =  -9154.166
Iteration 5:  Log likelihood = -9152.2717
Iteration 6:  Log likelihood =  -9152.244
Iteration 7:  Log likelihood =  -9152.244
Nominal response model                                  Number of obs = 1,799
Log likelihood = -9152.244
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Discrim						
2 vs 1	.4349366	.1221761	3.56	0.000	.1954759	.6743973
3 vs 1	-.5492982	.1630795	-3.37	0.001	-.8689282	-.2296683
4 vs 1	-.5967016	.1459728	-4.09	0.000	-.882803	-.3106002
Diff						
2 vs 1	-.8531528	.3228014	-2.64	0.008	-1.485832	-.2204737
3 vs 1	-1.346445	.3511587	-3.83	0.000	-2.034703	-.6581867
4 vs 1	-.5634533	.1473554	-3.82	0.000	-.8522645	-.2746422
q2						
Discrim						
2 vs 1	-.2226277	.2065114	-1.08	0.281	-.6273825	.1821272
3 vs 1	.6076979	.2000752	3.04	0.002	.2155576	.9998381
4 vs 1	-.1144097	.194821	-0.59	0.557	-.4962518	.2674324
Diff						
2 vs 1	5.169253	5.017017	1.03	0.303	-4.663919	15.00242
3 vs 1	-3.20606	1.043754	-3.07	0.002	-5.251781	-1.160339
4 vs 1	14.36817	24.79236	0.58	0.562	-34.22397	62.9603
q3						
Discrim						
2 vs 1	-.214686	.214803	-1.00	0.318	-.6356921	.2063201
3 vs 1	.7354083	.2053428	3.58	0.000	.3329438	1.137873
4 vs 1	1.272605	.2287827	5.56	0.000	.8241987	1.72101
Diff						
2 vs 1	1.3132	2.006775	0.65	0.513	-2.620008	5.246407
3 vs 1	-1.752087	.3919143	-4.47	0.000	-2.520224	-.9839486
4 vs 1	-1.145029	.177832	-6.44	0.000	-1.493573	-.7964847
q4						
Discrim						
2 vs 1	.9090316	.1734278	5.24	0.000	.5691194	1.248944
3 vs 1	.6275533	.1588602	3.95	0.000	.3161931	.9389136
4 vs 1	1.387606	.2034136	6.82	0.000	.9889224	1.786289
Diff						
2 vs 1	-.9150171	.1377816	-6.64	0.000	-1.185064	-.6449703
3 vs 1	-1.445922	.269392	-5.37	0.000	-1.97392	-.9179229
4 vs 1	-.4153831	.0870474	-4.77	0.000	-.5859929	-.2447734

Looking at item q1, we see that the second category, labeled 2 vs 1, is the most discriminating among the respondents. The difficulty parameters represent the points at which the base outcome intersects with the other outcomes. For item q1, the estimated difficulty for category 2 is -0.85; this means that a person with $\theta = -0.85$ would be equally likely to select response 1 or response 2 on this item. Likewise, category 3 has a estimated difficulty of -1.35, so a person with $\theta = -1.35$ would be equally likely to select responses 1 or 3.

For an NRM, it is easiest to interpret the parameters by plotting the CCCs. The curves trace the probability of choosing each category as a function of θ using the estimated NRM parameters. Here we plot the probabilities for item q1 using irtgraph icc; see [IRT] **irtgraph icc** for details.

```
. irtgraph icc q1, xlabel(-4 -1.35 -.85  -.56 4, alt)
```



We see that respondents with the latent trait level below approximately -0.7 tend to endorse category 4, and respondents with the latent trait level above that point tend to choose category 2.

Although the other two alternatives are dominated by category 2 and 4, this does not mean that the dominated alternatives are not viable responses. In the discussion above, we purposefully used "tend" rather than "most probable" because we can consider the probability of choosing category 4 in two ways: the probability of choosing category 4 over any other single category and the probability of choosing category 4 over all other categories combined. For values of the latent trait below about -0.7, the probability of choosing category 4 is larger than the probability of choosing any other individual category. However, for values of θ below approximately -2.7, we can also say that the probability of choosing category 4 is greater than the probability of choosing all other categories combined. In the range of $(-2.7, -0.7)$, the sum of the probabilities of categories 1, 2, and 3 together exceeds the probability of a response in category 4. Thus, in this range, it is more probable that a respondent chooses "anything other than 4" rather than "4". We can use the same argument for category 2 in the range $(-0.7, 0.6)$.

◁

Video example

Item response theory using Stata: Nominal response models (NRMs)

Stored results

irt nrm 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(k_out#)	number of outcomes for the #th item, nominal
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)	nrm
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(out#)	outcomes for the #th item, nominal
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. Without loss of generality, we will assume all items take on the unordered categories, $k = 1, \ldots, K$.

Using the IRT parameterization, we see that the probability of person j with latent trait level θ_j (the latent trait) providing response k for item i is given by

$$\Pr(Y_{ij} = k|\mathbf{a}_i, \mathbf{b}_i, \theta_j) = \frac{\exp\{a_{ik}(\theta_j - b_{ik})\}}{\sum_{h=1}^{K} \exp\{a_{ih}(\theta_j - b_{ih})\}}$$

where $\mathbf{a}_i = (a_{i1}, \ldots, a_{iK})$ represent the discrimination for each category of item i and $\mathbf{b}_i = (b_{i1}, \ldots, b_{iK})$ represent the difficulties for each category of item i. irt nrm fits the model using the slope-intercept form, so the probability for providing response k is parameterized as

$$\Pr(Y_{ij} = k|\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \theta_j) = \frac{\exp(\alpha_{ik}\theta_j + \beta_{ik})}{\sum_{h=1}^{K} \exp(\alpha_{ih}\theta_j + \beta_{ih})}$$

The transformation between these two parameterizations is

$$a_{ik} = \alpha_{ik} \qquad b_{ik} = -\frac{\beta_{ik}}{\alpha_{ik}}$$

irt nrm uses baseline constraints to ensure the model is identified; the baseline constraints are set on the slope and intercept for the first outcome for each item, for example, $a_{i1} = 0$ and $b_{i1} = 0$. This baseline outcome is necessary for the model to be identified.

Let y_{ij} be the observed response for Y_{ij} and $p_{ij} = \Pr(Y_{ij} = y_{ij}|\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \theta_j)$. Conditionally on θ_j, the item responses are assumed independent, so the conditional density for person j is given by

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

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_I, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_I)$, 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

Baker, F. B., and S.-H. Kim. 2004. *Item Response Theory: Parameter Estimation Techniques*. 2nd ed, revised and expanded. Boca Raton, FL: CRC Press.

Bock, R. D. 1972. Estimating item parameters and latent ability when responses are scored in two or more nominal categories. *Psychometrika* 37: 29–51. https://doi.org/10.1007/BF02291411.

de Ayala, R. J. 2022. *The Theory and Practice of Item Response Theory*. 2nd ed. New York: Guilford Press.

Also see

[IRT] **irt nrm postestimation** — Postestimation tools for irt nrm

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

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

[IRT] **irt grm** — Graded response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

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

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

[U] 20 Estimation and postestimation commands

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt nrm`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* #`)` may also be specified as `outcome(`#.*item*`)` or `outcome(`*item* `##`). `outcome(`*item* `#3`) means the third outcome value. `outcome(`*item* `#3`) would mean the same as `outcome(`*item* `4`) if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(`#`)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(`#`)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(`#`)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt nrm**.

By default, or when the `marginal` option is specified, the predicted probability for item i, person j, and outcome k is computed as

$$\hat{p}_{ijk} = \int_{-\infty}^{\infty} \Pr(Y_{ij} = k|\widehat{\boldsymbol{\alpha}}_i, \widehat{\boldsymbol{\beta}}_i, \theta_j)\phi(\theta_j)\, d\theta_j$$

where $\widehat{\boldsymbol{\alpha}}_i$ and $\widehat{\boldsymbol{\beta}}_i$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ijk} = \hat{\alpha}_{ik}\,\tilde{\theta}_j + \hat{\beta}_{ik}$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ijk} = \hat{\beta}_{ik}$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ijk} = \Pr(Y_{ij} = k|\widehat{\boldsymbol{\alpha}}_i, \widehat{\boldsymbol{\beta}}_i, \tilde{\theta}_j)$$

Also see

[IRT] **irt nrm** — Nominal response model

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt pcm — Partial credit model

Description

`irt pcm` fits partial credit models (PCMs) to ordinal items. In the PCM, items vary in their difficulty but share the same discrimination parameter.

`irt gpcm` fits generalized partial credit models (GPCMs) to ordinal items. In the GPCM, items vary in their difficulty and discrimination.

Quick start

PCM for ordinal items `o1` to `o5`

```
irt pcm o1-o5
```

Plot CCCs for `o1`

```
irtgraph icc o1
```

Menu

Statistics > IRT (item response theory)

Syntax

Partial credit model

```
irt pcm varlist [if] [in] [weight] [, options]
```

Generalized partial credit model

```
irt gpcm 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

The following discussion is about how to use `irt` to fit PCMs and GPCMs to ordinal items. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

The PCM is used for ordered categorical responses. An item scored $0, 1, \ldots, K$ is divided into K adjacent logits, and a positive response in category k implies a positive response to the categories preceding category k.

The probability of person j scoring in category k on item i is

$$\Pr(Y_{ij} = k|\theta_j) = \frac{\exp\{\sum_{t=1}^{k} a(\theta_j - b_{it})\}}{1 + \sum_{s=1}^{K} \exp\{\sum_{t=1}^{s} a(\theta_j - b_{it})\}} \qquad \theta_j \sim N(0,1)$$

where a represents the discrimination common to all items, b_{it} represents the difficulty that distinguishes outcome t from the other outcomes in item i, and θ_j is the latent trait of person j.

In a GPCM, each item has its own discrimination parameter.

The PCM was proposed by Masters (1982). The GPCM was proposed by Muraki (1992).

▷ Example 1: Fitting a PCM

To illustrate the PCM, we use the analogical reasoning data from de Ayala (2022). `alike.dta` contains eight questions, `v1` through `v8`, that ask how two things are alike, for example, "In what way are a dog and a lion alike?" Each response is graded as 0 (incorrect), 1 (partially correct), and 2 (correct). Here we list the first five observations.

```
. use https://www.stata-press.com/data/r19/alike
(Analogical reasoning data from de Ayala (2009))
. list in 1/5, nolabel
```

	v1	v2	v3	v4	v5	v6	v7	v8
1.	2	2	0	0	0	0	0	0
2.	2	0	2	1	2	1	0	0
3.	2	2	1	2	2	1	0	0
4.	2	2	2	1	2	0	0	0
5.	2	2	2	2	1	2	2	2

Looking across the first row, we see that the first respondent correctly solved items `v1` and `v2` and was incorrect on the remaining items.

We fit a PCM as follows:

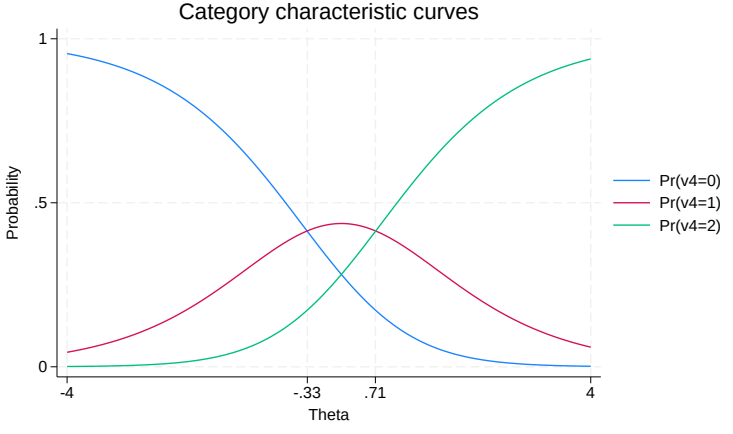
```
. irt pcm v1-v8
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -20869.947
Iteration 1:  Log likelihood = -20869.947  (backed up)
Fitting full model:
Iteration 0:  Log likelihood = -20048.975
Iteration 1:  Log likelihood = -19814.317
Iteration 2:  Log likelihood = -19678.395
Iteration 3:  Log likelihood = -19678.271
Iteration 4:  Log likelihood = -19678.271
Partial credit model                                  Number of obs = 2,941
Log likelihood = -19678.271
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.8375472	.0194059	43.16	0.000	.7995124	.875582
v1						
Diff						
1 vs 0	-1.546962	.1128848	-13.70	0.000	-1.768212	-1.325712
2 vs 1	-2.463391	.0900475	-27.36	0.000	-2.639881	-2.286902
v2						
Diff						
1 vs 0	-.508318	.0803871	-6.32	0.000	-.6658738	-.3507622
2 vs 1	-1.592003	.0753361	-21.13	0.000	-1.739659	-1.444347
v3						
Diff						
1 vs 0	-1.242774	.0719814	-17.27	0.000	-1.383855	-1.101694
2 vs 1	-.2770088	.0562749	-4.92	0.000	-.3873056	-.166712
v4						
Diff						
1 vs 0	-.3337874	.0580143	-5.75	0.000	-.4474934	-.2200814
2 vs 1	.7146057	.0614175	11.64	0.000	.5942296	.8349819
v5						
Diff						
1 vs 0	1.89372	.0969163	19.54	0.000	1.703768	2.083672
2 vs 1	-1.454011	.0955847	-15.21	0.000	-1.641353	-1.266668
v6						
Diff						
1 vs 0	-.2165156	.052177	-4.15	0.000	-.3187806	-.1142506
2 vs 1	3.115386	.1146119	27.18	0.000	2.89075	3.340021
v7						
Diff						
1 vs 0	1.909344	.0834947	22.87	0.000	1.745698	2.072991
2 vs 1	-.0129814	.0832004	-0.16	0.876	-.1760511	.1500883
v8						
Diff						
1 vs 0	1.514291	.0685158	22.10	0.000	1.380003	1.64858
2 vs 1	1.63067	.0933511	17.47	0.000	1.447705	1.813635

The difficulties represent a point at which the two adjacent categories are equally likely. For item v4, a person with $\theta = -0.33$ is equally likely to answer incorrectly or to answer partially correct (labeled 1 vs 0). A person with $\theta = 0.71$ is equally likely to be partially correct or to be correct (labeled 2 vs 1).

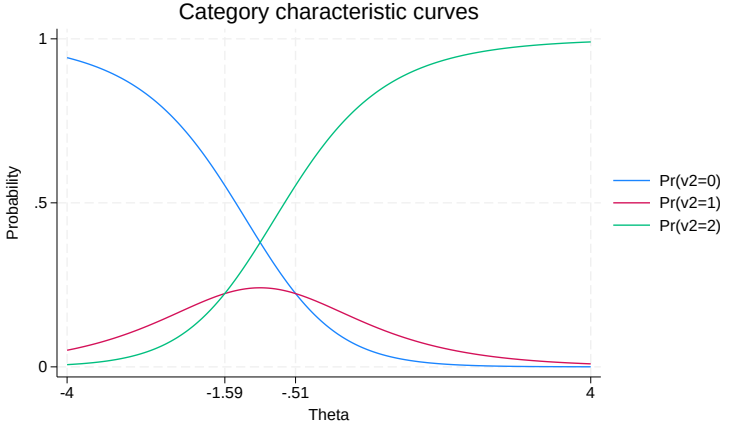
We can present this graphically using CCCs. The curves trace the probability of choosing each category as a function of θ using the estimated PCM parameters. Here we plot the probabilities for item v4 using `irtgraph icc`; see [IRT] **irtgraph icc** for details.

```
. irtgraph icc v4, xlabel(-4 -.33 .71 4)
```



While the PCM is intended for items having ordered categorical responses, the model is parameterized as if the outcomes were nominal. Therefore, the difficulty parameters for a given item are not necessarily in an increasing order. For example, for item v2, the second difficulty parameter is -1.59 and is smaller than the first difficulty parameter, -0.51. This is called a reversal and indicates that the category with the reversed threshold is dominated by the other two categories. Here we show this situation graphically.

```
. irtgraph icc v2, xlabel(-4 -.51 -1.59 4)
```



Notice that the probability of responding with a partially correct answer is never greater than both the probability of responding incorrectly and the probability of responding correctly. A reversal of the thresholds may indicate a potential problem with the item or with how raters graded the responses to the item. In our case, item v2 is primarily behaving like a binary item.

◁

Stored results

irt pcm and irt gpcm store 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(k_out#)	number of categories for the #th item, ordinal
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)	pcm or gpcm
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(out#)	categories for the #th item, ordinal
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. Without loss of generality, we will assume all items take on the ordered categories, $k = 0, 1, \ldots, K$.

Using the IRT parameterization, we see that the probability of person j with latent trait level θ_j (the latent trait) providing response k for item i is given by

$$\Pr(Y_{ij} = k|a_i, \mathbf{b}_i, \theta_j) = \frac{\exp\{\sum_{t=1}^{k} a_i(\theta_j - b_{it})\}}{1 + \sum_{s=1}^{K} \exp\{\sum_{t=1}^{s} a_i(\theta_j - b_{it})\}}$$

where a_i represents the discrimination for item i, $\mathbf{b}_i = (b_{i1}, \ldots, b_{iK})$ represent the difficulties that distinguish the ordered categories of item i, and it is understood that

$$\Pr(Y_{ij} = 0|a_i, \mathbf{b}_i, \theta_j) = \frac{1}{1 + \sum_{s=1}^{K} \exp\{\sum_{t=1}^{s} a_i(\theta_j - b_{it})\}}$$

irt pcm and irt gpcm fit the model using the slope-intercept form, so the probability for providing response k is parameterized as

$$\Pr(Y_{ij} = k|\alpha_i, \boldsymbol{\beta}_i, \theta_j) = \frac{\exp(k\alpha_i\theta_j + \beta_{ik})}{1 + \sum_{s=1}^{K} \exp(s\alpha_i\theta_j + \beta_{is})}$$

The transformation between these two parameterizations is

$$a_i = \alpha_i \qquad b_{ik} = -\frac{\beta_{ik} - \beta_{i,k-1}}{\alpha_i}$$

where $b_{i0} = 0$ and $\beta_{i0} = 0$. For irt pcm, the item discriminations a_i are constrained to be equal.

Let y_{ij} be the observed response for Y_{ij} and $p_{ij} = \Pr(Y_{ij} = y_{ij}|\alpha_i, \boldsymbol{\beta}_i, \theta_j)$. Conditional on θ_j, the item responses are assumed to be independent, so the conditional density for person j is given by

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

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha_1, \ldots, \alpha_I, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_I)$, 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

de Ayala, R. J. 2022. *The Theory and Practice of Item Response Theory*. 2nd ed. New York: Guilford Press.

Hamel, J.-F., G. Challet-Bouju, V. Sébille, and J.-B. Hardouin. 2016. Partial credit model: Estimations and tests of fit with pcmodel. *Stata Journal* 16: 464–481.

Masters, G. N. 1982. A Rasch model for partial credit scoring. *Psychometrika* 47: 149–174. https://doi.org/10.1007/BF02296272.

Muraki, E. 1992. A generalized partial credit model: Application of an EM algorithm. *Applied Psychological Measurement* 16: 159–176. https://doi.org/10.1177/014662169201600206.

Also see

[IRT] **irt pcm postestimation** — Postestimation tools for irt pcm

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

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

[IRT] **irt grm** — Graded response model

[IRT] **irt rsm** — Rating scale model

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

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

[U] 20 Estimation and postestimation commands

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt pcm` and `irt gpcm`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* *#*`)` may also be specified as `outcome(`*#.item*`)` or `outcome(`*item* `#`*#*`)`. `outcome(`*item* `#3``)` means the third outcome value. `outcome(`*item* `#3``)` would mean the same as `outcome(`*item* `4``)` if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(#)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(#)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(#)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt pcm**.

When the `marginal` option is specified, the predicted probability for item i, person j, and outcome k is computed as

$$\hat{p}_{ijk} = \int_{-\infty}^{\infty} \Pr(Y_{ij} = k|\hat{\alpha}_i, \widehat{\boldsymbol{\beta}}_i, \theta_j)\phi(\theta_j)\, d\theta_j$$

where $\hat{\alpha}_i$ and $\widehat{\boldsymbol{\beta}}_i$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ijk} = k\hat{\alpha}_i\, \tilde{\theta}_j + \hat{\beta}_{ik}$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ijk} = \hat{\beta}_{ik}$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ijk} = \Pr(Y_{ij} = k|\hat{\alpha}_i, \widehat{\boldsymbol{\beta}}_i, \tilde{\theta}_j)$$

Also see

[IRT] **irt pcm** — Partial credit model

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt rsm — Rating scale model

Description

`irt rsm` fits rating scale models (RSMs) to ordinal items. In the RSM, items vary in their difficulty but share the same discrimination parameter. The distances between the difficulties of adjacent outcomes are equal across the items.

Quick start

RSM for ordinal items `o1` to `o5`

```
irt rsm o1-o5
```

Plot CCCs for `o1`

```
irtgraph icc o1
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irt rsm 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

Remarks are presented under the following headings:

Overview

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

The RSM is a more parsimonious version of the PCM; see [IRT] **irt pcm**. In an RSM, the distances between categories are equal across all items.

The RSM is used for ordered categorical responses. An item scored $0, 1, \ldots, K$ is divided into K adjacent logits, and a positive response in category k implies a positive response to the categories preceding category k.

The probability of person j scoring in category k on item i is

$$\Pr(Y_{ij} = k|a, b_i, \mathbf{d}, \theta_j) = \frac{\exp[\sum_{t=1}^{k} a\{\theta_j - (b_i + d_t)\}]}{1 + \sum_{s=1}^{K} \exp[\sum_{t=1}^{s} a\{\theta_j - (b_i + d_t)\}]} \qquad \theta_j \sim N(0, 1)$$

where a represents the discrimination common to all items, b_i represents the "overall" difficulty of item i, $\mathbf{d} = (d_1, \ldots, d_K)$, d_t represents the threshold of outcome t common to all items such that $\sum_{t=1}^{K} d_t = 0$, and θ_j is the latent trait of person j.

Because all the items share the common thresholds, the difference between the difficulty parameters between adjacent categories is equal across the items. The presence of common thresholds requires that all items have the same number of responses. The responses are assumed to be functionally equivalent; that is, the responses should have the same meaning across all items.

The RSM was proposed by Andrich (1978a, 1978b).

Example 1: Fitting an RSM

To illustrate the RSM, we use the data from Zheng and Rabe-Hesketh (2007). `charity.dta` contains five survey questions, `ta1` through `ta5`, measuring faith and trust in charity organizations. Each item is coded 0, 1, 2, or 3, with higher scores indicating less favorable feelings toward charities.

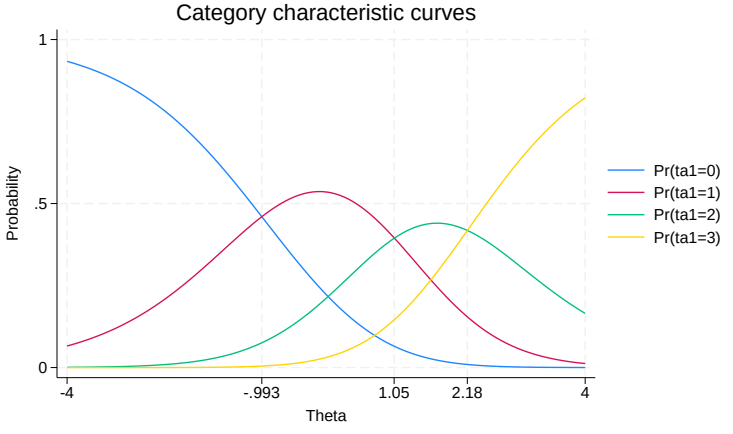
```
. use https://www.stata-press.com/data/r19/charity
(Data from Zheng & Rabe-Hesketh (2007))
. irt rsm ta1-ta5
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -5980.8848
Iteration 1:  Log likelihood = -5564.0205
Iteration 2:  Log likelihood = -5550.1989
Iteration 3:  Log likelihood = -5550.1765
Iteration 4:  Log likelihood = -5550.1765
Fitting full model:
Iteration 0:  Log likelihood = -5426.9653
Iteration 1:  Log likelihood = -5357.5172
Iteration 2:  Log likelihood = -5294.5245
Iteration 3:  Log likelihood = -5293.9321
Iteration 4:  Log likelihood = -5293.9307
Iteration 5:  Log likelihood = -5293.9307
Rating scale model                                      Number of obs = 945
Log likelihood = -5293.9307
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.8826766	.0416351	21.20	0.000	.8010734	.9642798
ta1						
Diff						
1 vs 0	-.9930361	.0787401	-12.61	0.000	-1.147364	-.8387083
2 vs 1	1.054185	.0819193	12.87	0.000	.8936264	1.214744
3 vs 2	2.180982	.1150909	18.95	0.000	1.955408	2.406556
ta2						
Diff						
1 vs 0	-1.640008	.0904366	-18.13	0.000	-1.81726	-1.462756
2 vs 1	.4072134	.0731437	5.57	0.000	.2638544	.5505725
3 vs 2	1.534011	.0988783	15.51	0.000	1.340213	1.727809
ta3						
Diff						
1 vs 0	-.9265681	.0767494	-12.07	0.000	-1.076994	-.776142
2 vs 1	1.120653	.0824001	13.60	0.000	.959152	1.282155
3 vs 2	2.24745	.1162845	19.33	0.000	2.019537	2.475364
ta4						
Diff						
1 vs 0	-.2352774	.0712757	-3.30	0.001	-.3749753	-.0955795
2 vs 1	1.811944	.0998673	18.14	0.000	1.616208	2.00768
3 vs 2	2.938741	.1355148	21.69	0.000	2.673137	3.204345
ta5						
Diff						
1 vs 0	-1.077613	.0791414	-13.62	0.000	-1.232728	-.9224992
2 vs 1	.9696079	.0796777	12.17	0.000	.8134425	1.125773
3 vs 2	2.096405	.1124727	18.64	0.000	1.875963	2.316848

The difficulties represent a point at which the two adjacent categories are equally likely. For item `ta1`, a person with $\theta = -0.993$ is equally likely to respond with a 0 or a 1, a person with $\theta = 1.05$ is equally likely to respond with a 1 or a 2, and a person with $\theta = 2.18$ is equally likely to respond with a 2 or a 3.

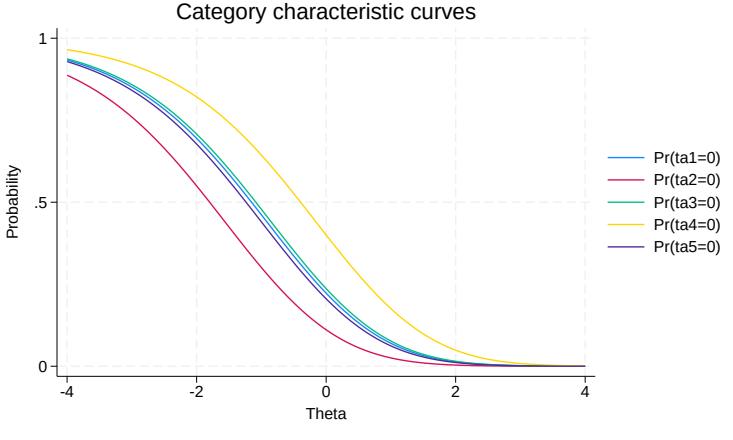
We can show this graphically using CCCs. The curves trace the probability of choosing each category as a function of θ using the estimated RSM parameters. Here we plot the probabilities for item `ta1` using `irtgraph icc`; see [IRT] **irtgraph icc** for details.

```
. irtgraph icc ta1, xlabel(-4 -.993 1.05 2.18 4)
```



Note that in the preceding estimation output, the distance between the estimated difficulties labeled `1 vs 0` and `2 vs 1` is the same for all items, and the same relationship holds for the distance between the estimated difficulties labeled `2 vs 1` and `3 vs 2`. Because of this, CCCs for all items have the same shape but are offset by a constant from each other. To see this graphically, we specify `0.ta*`, requesting that the CCC for the first category be shown for all items. The interested reader can create similar graphs for the other three categories to verify our claim.

```
. irtgraph icc 0.ta*
```



Video example

Item response theory using Stata: Rating scale models (RSMs)

Stored results

irt rsm 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(k_out#)	number of categories for the #th item, ordinal
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)	rsm
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(out#)	categories for the #th item, ordinal
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. Because of the constraints identified with this model, the RSM requires that all items take on the same number of ordered categories. Without loss of generality, we assume those categories are $k = 0, 1, \ldots, K$.

Using the IRT parameterization, we see that the probability of person j with latent trait level θ_j (the latent trait) providing response k for item i is given by

$$\Pr(Y_{ij} = k|a, b_i, \mathbf{d}, \theta_j) = \frac{\exp[\sum_{t=1}^{k} a\{\theta_j - (b_i + d_t)\}]}{1 + \sum_{s=1}^{K} \exp[\sum_{t=1}^{s} a\{\theta_j - (b_i + d_t)\}]}$$

where a represents the discrimination, b_i represents the overall difficulty of item i, $\mathbf{d} = (d_1, \ldots, d_K)$ represent the thresholds, common to all items, that separate adjacent response categories, and it is understood that

$$\Pr(Y_{ij} = 0|a, b_i, \mathbf{d}, \theta_j) = \frac{1}{1 + \sum_{s=1}^{K} \exp[\sum_{t=1}^{s} a\{\theta_j - (b_i + d_t)\}]}$$

irt rsm fits the model using the slope-intercept form, so the probability for providing response k is parameterized as

$$\Pr(Y_{ij} = k|\alpha, \beta_i, \boldsymbol{\delta}, \theta_j) = \frac{\exp(k\alpha\theta_j + k\beta_i + \delta_k)}{1 + \sum_{s=1}^{K} \exp(s\alpha\theta_j + s\beta_i + \delta_s)}$$

The transformation between these two parameterizations is

$$a = \alpha \qquad b_i = -\frac{\beta_i}{\alpha} \qquad d_t = -\frac{(\delta_t - \delta_{t-1})}{\alpha}$$

where $d_0 = 0$ and $\delta_0 = 0$. Because the thresholds are common to all items, irt rsm requires the items must all take on the same number of ordered categories.

Let y_{ij} be the observed response for Y_{ij} and $p_{ij} = \Pr(Y_{ij} = y_{ij}|\alpha, \beta_i, \boldsymbol{\delta}, \theta_j)$. Conditional on θ_j, the item responses are assumed to be independent, so the conditional density for person j is given by

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

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\alpha, \beta_1, \ldots, \beta_I, \delta_1, \ldots, \delta_K)$, I is the number of items, and K is the number of response categories.

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

Andrich, D. 1978a. Application of a psychometric rating model to ordered categories which are scored with successive integers. *Applied Psychological Measurement* 2: 581–594. https://doi.org/10.1177/014662167800200413.

———. 1978b. A rating formulation for ordered response categories. *Psychometrika* 43: 561–573. https://doi.org/10.1007/BF02293814.

Zheng, X., and S. Rabe-Hesketh. 2007. Estimating parameters of dichotomous and ordinal item response models with gllamm. *Stata Journal* 7: 313–333.

Also see

[IRT] **irt rsm postestimation** — Postestimation tools for irt rsm

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

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

[IRT] **irt pcm** — Partial credit model

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

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

[U] 20 Estimation and postestimation commands

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt rsm`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* *#*`)` may also be specified as `outcome(`*#.item*`)` or `outcome(`*item* `#`*#*`)`. `outcome(`*item* `#3)` means the third outcome value. `outcome(`*item* `#3)` would mean the same as `outcome(`*item* `4)` if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(`#`)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(`#`)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(`#`)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

Empirical Bayes predictions of the latent trait are documented in *Methods and formulas* of [IRT] **irt hybrid postestimation**.

This section builds on the notation introduced in *Methods and formulas* of [IRT] **irt rsm**.

When the `marginal` option is specified, the predicted probability for item i, person j, and outcome k is computed as

$$\hat{p}_{ijk} = \int_{-\infty}^{\infty} \Pr(Y_{ij} = k|\hat{\alpha}, \hat{\beta}_i, \hat{\boldsymbol{\delta}}, \theta_j)\phi(\theta_j)\, d\theta_j$$

where $\hat{\alpha}$, $\hat{\beta}_i$, and $\hat{\boldsymbol{\delta}}$ are the estimated parameters in the slope-intercept parameterization. The integral is approximated using standard Gauss–Hermite quadrature.

In what follows, we show formulas using the posterior means estimates of latent trait $\tilde{\theta}_j$, which are computed by default or when the `conditional(ebmeans)` option is specified. If the `conditional(ebmodes)` option is specified, $\tilde{\theta}_j$ are simply replaced with the posterior modes $\tilde{\tilde{\theta}}_j$ in these formulas.

For the response to item i from person j, the linear predictor is computed as

$$\hat{z}_{ijk} = k\hat{\alpha}\,\tilde{\theta}_j + k\hat{\beta}_i + \hat{\delta}_k$$

If option `marginal` or `conditional(fixedonly)` is specified, the linear predictor is computed as

$$\hat{z}_{ijk} = k\hat{\beta}_i + \hat{\delta}_k$$

The predicted probability, conditional on the predicted latent trait, is

$$\hat{p}_{ijk} = \Pr(Y_{ij} = k|\hat{\alpha}, \hat{\beta}_i, \hat{\boldsymbol{\delta}}, \tilde{\theta}_j)$$

Also see

[IRT] **irt rsm** — Rating scale model

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

irt hybrid — Hybrid IRT models

Description

`irt hybrid` fits IRT models to combinations of binary, ordinal, and nominal items.

Quick start

1PL model for binary items `b1` to `b5` and 2PL model for binary items `b6` to `b10`

```
irt hybrid (1pl b1-b5) (2pl b6-b10)
```

Plot ICCs for each item

```
irtgraph icc
```

GRM for ordinal items `o1`, `o2`, and `o3`, 3PL model for binary items `b1` and `b2`, and NRM for nominal items `n1` to `n5`

```
irt hybrid (grm o1 o2 o3) (3pl b1 b2) (nrm n1-n5)
```

Plot CCCs for `o1`

```
irtgraph icc o1
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irt hybrid (model varlist1 [ , mopts ]) (model varlist2 [ , mopts ]) [ ... ]
    [ if ] [ in ] [ weight ] [ , options ]
```

model	Description
1pl	One-parameter logistic model
2pl	Two-parameter logistic model
3pl	Three-parameter logistic model
grm	Graded response model
pcm	Partial credit model
gpcm	Generalized partial credit model
rsm	Rating scale model
nrm	Nominal response model

mopts	Description
cns(*spec*)	apply specified parameter constraints
sepguessing	estimate a separate pseudoguessing parameter for each item; allowed only with a 3pl model
gsepguessing	estimate separate pseudoguessing parameters for each group; allowed only with a group 3pl model

options	Description
group(*varname*)	fit model for different groups
Model	
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.

mopts

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

sepguessing specifies that a separate pseudoguessing parameter be estimated for each item. This option is allowed only with a 3pl model; see [IRT] **irt 3pl** for details.

gsepguessing specifies that separate pseudoguessing parameters be estimated for each group. This option is allowed only with a group 3pl model.

Options

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

Model

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

The following discussion is about how to use irt to fit hybrid IRT models. In a hybrid model, one can fit different IRT models to subsets of items and perform a single calibration for the whole instrument. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first. If you are interested in the details of a specific IRT model, we refer you to the following.

Binary response models

irt 1pl	One-parameter logistic model
irt 2pl	Two-parameter logistic model
irt 3pl	Three-parameter logistic model

Categorical response models

irt grm	Graded response model
irt nrm	Nominal response model
irt pcm	Partial credit model
irt rsm	Rating scale model

▷ Example 1: Combining an NRM and a PCM within a single instrument

In example 1 of [IRT] **irt nrm**, we applied a NRM to the physical science test data from de Ayala (2022). The last item is in fact an open-ended question scored on a scale of 1 to 4; thus, a PCM may be more appropriate for this item.

We fit an NRM to items q1–q3 and a PCM to item q4 as follows:

```
. use https://www.stata-press.com/data/r19/science
(Physical science data from de Ayala (2009))
. irt hybrid (nrm q1-q3) (pcm q4)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -9256.1514
Iteration 1:  Log likelihood = -9256.1514
Fitting full model:
Iteration 0:  Log likelihood = -9383.3438  (not concave)
Iteration 1:  Log likelihood = -9251.6343  (not concave)
Iteration 2:  Log likelihood = -9200.7879
Iteration 3:  Log likelihood = -9183.2288
Iteration 4:  Log likelihood = -9169.5042
Iteration 5:  Log likelihood = -9168.4031
Iteration 6:  Log likelihood = -9168.2855
Iteration 7:  Log likelihood = -9168.2854
```

```
Hybrid IRT model                                        Number of obs = 1,799
Log likelihood = -9168.2854
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
nrm						
q1						
Discrim						
2 vs 1	.4646853	.1344318	3.46	0.001	.2012039	.7281667
3 vs 1	-.7170683	.1835435	-3.91	0.000	-1.076807	-.3573297
4 vs 1	-.5973805	.1601828	-3.73	0.000	-.9113331	-.2834279
Diff						
2 vs 1	-.7692399	.3102269	-2.48	0.013	-1.377274	-.1612063
3 vs 1	-1.137519	.2353162	-4.83	0.000	-1.59873	-.6763076
4 vs 1	-.5488615	.1512123	-3.63	0.000	-.845232	-.2524909
q2						
Discrim						
2 vs 1	-.1940623	.2058631	-0.94	0.346	-.5975465	.2094219
3 vs 1	.5554902	.2001945	2.77	0.006	.1631162	.9478641
4 vs 1	-.0926769	.1957145	-0.47	0.636	-.4762702	.2909165
Diff						
2 vs 1	5.986206	6.587602	0.91	0.364	-6.925257	18.89767
3 vs 1	-3.529391	1.256803	-2.81	0.005	-5.99268	-1.066102
4 vs 1	17.80318	37.97769	0.47	0.639	-56.63173	92.23809
q3						
Discrim						
2 vs 1	-.2375187	.2286107	-1.04	0.299	-.6855875	.2105501
3 vs 1	.701352	.2196849	3.19	0.001	.2707775	1.131927
4 vs 1	1.274855	.2460035	5.18	0.000	.7926971	1.757013
Diff						
2 vs 1	1.128287	1.746527	0.65	0.518	-2.294843	4.551418
3 vs 1	-1.824874	.4596278	-3.97	0.000	-2.725728	-.9240203
4 vs 1	-1.131441	.1882566	-6.01	0.000	-1.500417	-.7624645
pcm						
q4						
Discrim	.3300808	.0526185	6.27	0.000	.2269504	.4332111
Diff						
2 vs 1	-2.056822	.322035	-6.39	0.000	-2.687999	-1.425644
3 vs 2	-.2976236	.1923535	-1.55	0.122	-.6746294	.0793823
4 vs 3	.8472731	.2048051	4.14	0.000	.4458626	1.248684

Note how the NRM and PCM are separated, so it is easy to tell which parameters correspond to which model. Because the PCM is nested in the NRM, we could perform a likelihood-ratio test to see whether our model is preferable to a pure NRM model; see example 1 in [IRT] **irt** and [R] **lrtest** for more information. ◁

▷ Example 2: The 3PL model revisited

In example 1 of [IRT] **irt 3pl**, we used the mathematics and science data from De Boeck and Wilson (2004) to fit a 3PL model where the pseudoguessing parameter was constrained to be the same across items q1–q9. We mentioned that model identification problems can occur when one tries to estimate a separate guessing parameter for each item. In this example, we show how to deal with identification problems by constraining some pseudoguessing parameters to zero and fitting a 3PL model with separate guessing parameters to the remaining items.

We first fit a full 3PL model to items q1–q9, where each item has its own pseudoguessing parameter. Because the corresponding fixed-effects model is not identified, we limit the number of iterations irt spends fitting the fixed-effects model to 5.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt 3pl q1-q9, sepguessing startvalues(iterate(5))
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -5322.8824
Iteration 1:  Log likelihood = -4291.3914  (not concave)
Iteration 2:  Log likelihood = -4270.0005  (not concave)
Iteration 3:  Log likelihood = -4269.7927  (not concave)
Iteration 4:  Log likelihood = -4269.7825  (not concave)
Iteration 5:  Log likelihood = -4269.7825  (not concave)
Fitting full model:
Iteration 0:  Log likelihood = -4227.4731  (not concave)
Iteration 1:  Log likelihood = -4188.8074  (not concave)
Iteration 2:  Log likelihood =  -4134.829  (not concave)
Iteration 3:  Log likelihood = -4121.9664  (not concave)
Iteration 4:  Log likelihood =  -4120.161  (not concave)
Iteration 5:  Log likelihood =   -4119.33
Iteration 6:  Log likelihood = -4118.0626
Iteration 7:  Log likelihood = -4117.0488
Iteration 8:  Log likelihood = -4115.6541
Iteration 9:  Log likelihood = -4115.4168
Iteration 10: Log likelihood = -4114.5522
Iteration 11: Log likelihood = -4114.3738
Iteration 12: Log likelihood = -4114.1039
Iteration 13: Log likelihood = -4113.9668
Iteration 14: Log likelihood = -4113.9036
Iteration 15: Log likelihood = -4113.7972  (not concave)
Iteration 16: Log likelihood = -4113.7712
Iteration 17: Log likelihood = -4113.7505
Iteration 18: Log likelihood = -4113.7226
Iteration 19: Log likelihood = -4113.7178
Iteration 20: Log likelihood = -4113.7089
Iteration 21: Log likelihood = -4113.7033
Iteration 22: Log likelihood = -4113.6975
Iteration 23: Log likelihood = -4113.6964
Iteration 24: Log likelihood = -4113.6948
Iteration 25: Log likelihood = -4113.6939
Iteration 26: Log likelihood = -4113.6936
Iteration 27: Log likelihood = -4113.6934
Iteration 28: Log likelihood = -4113.6934
Iteration 29: Log likelihood = -4113.6933
```

Three-parameter logistic model — Number of obs = 800
Log likelihood = -4113.6933

		Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1							
	Discrim	2.412093	.8686457	2.78	0.005	.709579	4.114608
	Diff	-.0919186	.2094281	-0.44	0.661	-.5023902	.318553
	Guess	.2054622	.1067005			-.003667	.4145914
q2							
	Discrim	.6595413	.1142254	5.77	0.000	.4356637	.883419
	Diff	-.14912	.1209748	-1.23	0.218	-.3862263	.0879862
	Guess	6.49e-06	.004595			-.0089996	.0090126
q3							
	Discrim	1.002138	.1672076	5.99	0.000	.6744172	1.329859
	Diff	-1.61108	.218446	-7.38	0.000	-2.039226	-1.182934
	Guess	9.89e-08	.0004223			-.0008275	.0008277
q4							
	Discrim	1.32466	.5435057	2.44	0.015	.2594084	2.389911
	Diff	.811461	.2349897	3.45	0.001	.3508897	1.272032
	Guess	.1921895	.0961314			.0037755	.3806035
q5							
	Discrim	.8519931	.1450072	5.88	0.000	.5677843	1.136202
	Diff	1.653157	.244456	6.76	0.000	1.174032	2.132282
	Guess	1.27e-08	.0001261			-.0002471	.0002471
q6							
	Discrim	2.160352	.8886889	2.43	0.015	.4185542	3.90215
	Diff	.8876168	.1201165	7.39	0.000	.6521929	1.123041
	Guess	.1725436	.0486371			.0772168	.2678705
q7							
	Discrim	.9442741	2.196661	0.43	0.667	-3.361102	5.24965
	Diff	2.599643	1.80189	1.44	0.149	-.9319954	6.131282
	Guess	.1862207	.2136893			-.2326026	.605044
q8							
	Discrim	1.477403	.2581921	5.72	0.000	.9713561	1.983451
	Diff	-1.664011	.1868776	-8.90	0.000	-2.030284	-1.297737
	Guess	1.33e-09	.0000219			-.0000429	.0000429
q9							
	Discrim	.6233966	.1200201	5.19	0.000	.3881615	.8586317
	Diff	-1.536892	.2867222	-5.36	0.000	-2.098858	-.9749272
	Guess	2.22e-08	.0001789			-.0003506	.0003506

We see that the pseudoguessing parameters for items q2, q3, q5, q8, and q9 are very close to zero. This suggests that we could fit a 2PL model to these five items and a full 3PL model with separate guessing parameters to the remaining four items.

```
. irt hybrid (2pl q2 q3 q5 q8 q9) (3pl q1 q4 q6 q7, sepg), startval(iter(5))
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -4846.1954
Iteration 1:  Log likelihood = -4274.9988  (not concave)
Iteration 2:  Log likelihood = -4269.8038  (not concave)
Iteration 3:  Log likelihood = -4269.7889  (not concave)
Iteration 4:  Log likelihood = -4269.7825  (not concave)
Iteration 5:  Log likelihood = -4269.7825  (not concave)
Fitting full model:
Iteration 0:  Log likelihood =   -4237.32  (not concave)
Iteration 1:  Log likelihood = -4156.6562
Iteration 2:  Log likelihood = -4122.4275
Iteration 3:  Log likelihood = -4115.0165
Iteration 4:  Log likelihood = -4113.7357
Iteration 5:  Log likelihood = -4113.7317
Iteration 6:  Log likelihood = -4113.7155  (not concave)
Iteration 7:  Log likelihood = -4113.7153
Iteration 8:  Log likelihood = -4113.7124
Iteration 9:  Log likelihood = -4113.7041
Iteration 10: Log likelihood = -4113.6966
Iteration 11: Log likelihood = -4113.6965
Iteration 12: Log likelihood = -4113.6938
Iteration 13: Log likelihood =  -4113.694
Iteration 14: Log likelihood = -4113.6933
Iteration 15: Log likelihood = -4113.6933
Hybrid IRT model                                        Number of obs = 800
Log likelihood = -4113.6933
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q2						
Discrim	.6595289	.1141777	5.78	0.000	.4357448	.8833131
Diff	-.1491663	.1200093	-1.24	0.214	-.3843802	.0860477
q3						
Discrim	1.002143	.1672015	5.99	0.000	.6744346	1.329852
Diff	-1.611069	.2184352	-7.38	0.000	-2.039194	-1.182944
q5						
Discrim	.8519284	.1449869	5.88	0.000	.5677594	1.136097
Diff	1.65315	.2444541	6.76	0.000	1.174029	2.132271
q8						
Discrim	1.477406	.2581479	5.72	0.000	.9714453	1.983366
Diff	-1.664009	.1868519	-8.91	0.000	-2.030232	-1.297786
q9						
Discrim	.6233934	.1200188	5.19	0.000	.3881608	.858626
Diff	-1.536899	.286721	-5.36	0.000	-2.098862	-.9749362

3pl

q1						
Discrim	2.4122	.868651	2.78	0.005	.7096758	4.114725
Diff	-.0918963	.2094073	-0.44	0.661	-.502327	.3185344
Guess	.2054735	.1066903			-.0036357	.4145828
q4						
Discrim	1.324646	.5435556	2.44	0.015	.2592968	2.389995
Diff	.8114595	.2349801	3.45	0.001	.350907	1.272012
Guess	.1921879	.0961354			.0037661	.3806098
q6						
Discrim	2.160387	.8880769	2.43	0.015	.4197886	3.900986
Diff	.8876139	.120097	7.39	0.000	.652228	1.123
Guess	.1725445	.0485985			.0772932	.2677957
q7						
Discrim	.9441699	2.16823	0.44	0.663	-3.305483	5.193823
Diff	2.599752	1.781284	1.46	0.144	-.8915	6.091004
Guess	.1862199	.2109529			-.2272402	.5996801

Looking at the output, we see that the guessing parameters for q1, q4, q6, and q7 are similar. So we could further simplify this model by constraining the pseudoguessing parameter to be the same for the 3PL items. We could use a series of likelihood-ratio tests to choose among the competing models; see example 1 in [IRT] **irt** and [R] **lrtest** for more information.

◁

Stored results

irt 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_items#)	number of items in #th IRT equation
e(sepguess#)	1 if #th IRT model contains a separate pseudoguessing parameter
e(k_cat#)	number of categories for the #th item, ordinal
e(k_out#)	number of outcomes for the #th item, nominal
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(model#)	name of IRT model for the #th equation

e(items#)	names of items in #th 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)	parameter vector
e(b_pclass)	parameter class
e(cat#)	categories for the #th item, ordinal
e(out#)	outcomes for the #th item, nominal
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	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

Methods and formulas are presented under the following headings:

The likelihood
Groups
Gauss–Hermite quadrature
Adaptive quadrature

The likelihood

Let y_{ij} be the observed outcome for item i from person j. Define $p_{ij} = \Pr(Y_{ij} = y_{ij}|\boldsymbol{B}_i, \theta_j)$, where Y_{ij} represents the (yet to be observed) outcome, $\boldsymbol{B}_i$ contains parameters for item i, and θ_j is the ability (the latent trait) of person j. The functional form of p_{ij} and the parameters that go into $\boldsymbol{B}_i$ depend on the choice of IRT model for item i.

Conditional on θ_j, the item responses are assumed to be independent, so the conditional density for person j is given by

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

where $\mathbf{y}_j = (y_{1j}, \ldots, y_{Ij})$, $\boldsymbol{B} = (\boldsymbol{B}_1, \ldots, \boldsymbol{B}_I)$, 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.

Groups

When the `group()` option is specified, each group has its own model parameters. The collection of model parameters is

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \\ \vdots \\ \boldsymbol{B}_G \end{pmatrix}$$

where G is the number of groups.

The overall log likelihood is

$$\log L(\boldsymbol{B}) = \sum_{g=1}^{G} \log L(\boldsymbol{B}_g)$$

Gauss–Hermite quadrature

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 (x_q^*, w_q^*), $q = 1, \ldots, Q$. The GHQ approximation is then

$$\int_{-\infty}^{\infty} f(x) \exp(-x^2)\, dx \approx \sum_{q=1}^{Q} w_q^* f(x_q^*)$$

Using the standard normal distribution yields the approximation

$$\int_{-\infty}^{\infty} f(x)\phi(x)\, dx \approx \sum_{q=1}^{Q} w_q f(x_q)$$

where $x_q = \sqrt{2}x_q^*$ and $w_q = w_q^*/\sqrt{\pi}$. The GHQ approximation to the likelihood for person j is

$$L_j^{\text{GHQ}}(\boldsymbol{B}) = \sum_{q=1}^{Q} w_q f(\mathbf{y}_j|\boldsymbol{B}, x_q)$$

Adaptive quadrature

This section sets the stage for mean–variance adaptive Gauss–Hermite quadrature (MVAGHQ) and mode-curvature adaptive Gauss–Hermite quadrature (MCAGHQ).

If we fix the item variables and the model parameters, we see that the posterior density for θ_j is proportional to

$$\phi(\theta_j) f(\mathbf{y}_j|\boldsymbol{B}, \theta_j)$$

It is reasonable to assume that this posterior density can be approximated by a normal density with mean μ_j and variance τ_j. Instead of using the prior density of θ_j as the weighting distribution in the integral, we can use our approximation for the posterior density,

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

The likelihood is then approximated with

$$L_j^*(\boldsymbol{B}) = \sum_{q=1}^{Q} \omega_q f(\mathbf{y}_j|\boldsymbol{B}, \xi_q)$$

where ξ_q and the ω_q are functions of x_q and w_q and the adaptive parameters μ_j and τ_j.

For MVAGHQ, μ_j is the posterior mean, and τ_j is the posterior variance of θ_j. They are computed iteratively by updating the posterior moments by using the MVAGHQ approximation, starting with a zero mean and unit variance.

For MCAGHQ, μ_j is the posterior mode for θ_j, and τ_j is the curvature at the mode. They are computed by optimizing the joint density with respect to θ_j.

References

de Ayala, R. J. 2022. *The Theory and Practice of Item Response Theory*. 2nd ed. New York: Guilford Press.

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Also see

[IRT] **irt hybrid postestimation** — Postestimation tools for irt hybrid

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

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

[IRT] **irt grm** — Graded response model

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

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

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

[U] 20 Estimation and postestimation commands

Postestimation commands predict Methods and formulas References Also see

Postestimation commands

The following postestimation commands are of special interest after `irt`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* *#*`)` may also be specified as `outcome(`*#.item*`)` or `outcome(`*item* `#`*#*`)`. `outcome(`*item* `#3``)` means the third outcome value. `outcome(`*item* `#3``)` would mean the same as `outcome(`*item* `4``)` if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

`ebmodes` specifies that empirical Bayes modes are used to predict the latent variables.

`se(`*newvar*`)` calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the `latent` option.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of the length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

Integration

`intpoints(`#`)` specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

`iterate(`#`)` specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

`tolerance(`#`)` specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

This section builds on the notation introduced in *Methods and formulas* of each of the other IRT postestimation entries.

We begin by considering the prediction of the latent trait θ for a given person. Prediction of the latent trait in IRT models involves assigning a value to the latent trait, 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 traits: empirical Bayes means (also known as posterior means) and empirical Bayes modes (also known as posterior modes).

Methods and formulas are presented under the following headings:

Empirical Bayes

Let $\widehat{\boldsymbol{B}}$ denote the estimated model parameters. Empirical Bayes (EB) predictors of the latent trait are the means or modes of the empirical posterior distribution with the parameter estimates $\boldsymbol{B}$ replaced with their estimates $\widehat{\boldsymbol{B}}$. The method is called "empirical" because $\widehat{\boldsymbol{B}}$ is treated as known. EB combines the prior information about the latent trait with the likelihood to obtain the conditional posterior distribution of the latent trait. Using Bayes's theorem, we see that the empirical conditional posterior distribution of the latent trait for person j is

$$\begin{aligned}\omega(\theta_j|\mathbf{y}_j;\widehat{\boldsymbol{B}}) &= \frac{f(\mathbf{y}_j|\widehat{\boldsymbol{B}},\theta_j)\,\phi(\theta_j)}{\int_{-\infty}^{\infty} f(\mathbf{y}_j|\widehat{\boldsymbol{B}},\theta_j)\,\phi(\theta_j)\,d\theta_j}\\ &= \frac{f(\mathbf{y}_j|\widehat{\boldsymbol{B}},\theta_j)\,\phi(\theta_j)}{L_j(\widehat{\boldsymbol{B}})}\end{aligned}$$

The denominator is just the likelihood contribution for person j.

EB mean predictions of the latent trait, also known as posterior means, are calculated as

$$\tilde{\theta}_j = \int_{-\infty}^{\infty} \theta_j \, \omega(\theta_j|\mathbf{y}_j; \widehat{\boldsymbol{B}}) \, d\theta_j$$

where we use the notation $\tilde{\theta}_j$ rather than $\hat{\theta}_j$ to distinguish predicted values from estimates. This integral is approximated by MVAGHQ.

EB modal predictions can be approximated by solving for $\tilde{\tilde{\theta}}_j$ such that

$$\frac{\partial}{\partial \theta_j} \log\omega(\theta_j|\mathbf{y}_j; \widehat{\boldsymbol{B}})\big|_{\theta_j=\tilde{\tilde{\theta}}_j} = \mathbf{0}$$

Because the denominator in $\omega(\cdot)$ does not depend on θ_j, 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 the normal distribution, EB modes get closer and closer to EB means.

Just as there are many methods of assigning values to the latent trait, there are many methods of calculating standard errors of the predicted latent trait; 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 the latent trait. The EB posterior variance of the latent trait is given by

$$\mathrm{Var}(\tilde{\theta}_j|\mathbf{y}_j; \widehat{\boldsymbol{B}}) = \int_{-\infty}^{\infty} (\theta_j - \tilde{\theta}_j)^2 \, \omega(\theta_j|\mathbf{y}_j; \widehat{\boldsymbol{B}}) \, d\theta_j$$

The posterior variance and the integrals are approximated by MVAGHQ.

Conditional standard errors for the estimated posterior modes are derived from the standard theory of maximum likelihood, which dictates that the asymptotic variance matrix of the posterior mode is the negative inverse of the Hessian matrix.

Other predictions

The other predictions are governed by the model selected for the specified item response variable. For binary items, see *Methods and formulas* of the postestimation entries for `irt 1pl`, `irt 2pl`, and `irt 3pl`. For ordered items, see *Methods and formulas* of the postestimation entries for `irt grm`, `irt pcm`, and `irt rsm`. For nominal items, see *Methods and formulas* of the postestimation entry for `irt nrm`.

References

Skrondal, A., and S. Rabe-Hesketh. 2004. *Generalized Latent Variable Modeling: Multilevel, Longitudinal, and Structural Equation Models*. Boca Raton, FL: Chapman and Hall/CRC.

———. 2009. Prediction in multilevel generalized linear models. *Journal of the Royal Statistical Society*, A ser., 172: 659–687. https://doi.org/10.1111/j.1467-985X.2009.00587.x.

Also see

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

Description

Multiple-group IRT models are combined models across groups of the data. They allow some parameters to vary across groups and constrain others to be equal. The groups could be males and females, age categories, and the like.

The `irt` commands fit multiple-group IRT models when the `group(`*varname*`)` option is specified.

Quick start

1PL model for binary items `b1` to `b9` with difficulty and discrimination parameters equal across groups

```
irt 1pl b1-b9, group(grpvar)
```

Same as above, but with difficulty and discrimination parameters allowed to differ across groups

```
irt (0: 1pl b1-b9) (1: 1pl b1-b9), group(grpvar)
```

Same as above, but with discrimination parameter (`a`) equal across groups

```
irt (0: 1pl b1-b9, cns(a@k)) (1: 1pl b1-b9, cns(a@k)), group(grpvar)
```

1PL model with different items administered across groups

```
irt (0: 1pl b1-b7) (1: 1pl b3-b9), group(grpvar)
```

Same as above, but with difficulty and discrimination of `b3-b7` equal across groups

```
irt (0: 1pl b1 b2) (1pl b3-b7) (1: 1pl b8 b9), group(grpvar)
```

Display parameter estimates in compact form

```
estat greport
```

Plot ICCs for item `b5` for both groups

```
irtgraph icc b5
```

Same as above, but change the line pattern for group 1

```
irtgraph icc (0: b5) (1: b5, lpattern(dash))
```

Menu

Statistics > IRT (item response theory)

Syntax

Single-equation syntax

```
irt model varlist ..., group(varname) [options]
```

Multiple-equation syntax

```
irt ([#:] model varlist1 [, mopts]) ([#:] model varlist2 [, mopts]) ...,
    group(varname) [options]
```

#: specifies the group for which *model* is to be fit.

model	Description
1pl	One-parameter logistic model
2pl	Two-parameter logistic model
3pl	Three-parameter logistic model
grm	Graded response model
pcm	Partial credit model
gpcm	Generalized partial credit model
rsm	Rating scale model
nrm	Nominal response model

options	Description
`group(`*varname*`)`	fit model for different groups
Model	
* `cns(`*spec*`)`	apply specified parameter constraints
`listwise`	drop observations with any missing items
* `sepguessing`	estimate a separate pseudoguessing parameter for each item
* `gsepguessing`	estimate separate pseudoguessing parameters for each group
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

**mopts* are `cns()`, `sepguessing`, and `gsepguessing`.

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

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

group(*varname*) specifies that the model parameters be allowed to vary across values of *varname*. *varname* might be sex, and then parameters may vary 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 constraining all item coefficients to be equal across groups in each model specified without a group identifier. The mean and the variance of the latent trait are constrained to 0 and 1, respectively, for the group corresponding to the smallest value of *varname* (reference group) and estimated for the remaining groups (focal groups).

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.

sepguessing specifies that a separate pseudoguessing parameter be estimated for each item. This option is allowed only with a 3pl model; see [IRT] **irt 3pl** for details.

gsepguessing specifies that separate pseudoguessing parameters be estimated for each group. This option is allowed only with a 3pl model.

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

Remarks are presented under the following headings:

Overview

The following discussion is about how to perform multiple-group analysis with irt. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

Multiple-group IRT analysis is usually performed when you believe one or more items function differently across groups; see [IRT] **DIF** for further details.

Baseline group model

▷ Example 1: Fitting a 2PL model for different groups of the data

To illustrate a multiple-group IRT 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. There are 761 male students and 739 female students.

```
. use https://www.stata-press.com/data/r19/masc2
(Data from De Boeck & Wilson (2004))
. tabulate female
```

Female	Freq.	Percent	Cum.
Male	761	50.73	50.73
Female	739	49.27	100.00
Total	1,500	100.00	

Say we are interested in measuring mathematical ability using binary items q1–q5. We fit a two-group 2PL model as follows:

```
. irt 2pl q1-q5, group(female)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -4594.5412
Iteration 1:  Log likelihood = -4590.4516
Iteration 2:  Log likelihood = -4590.4502
Iteration 3:  Log likelihood = -4590.4502
Group: Male
Group: Female
Fitting full model:
Iteration 0:  Log likelihood = -4503.5396  (not concave)
Iteration 1:  Log likelihood = -4479.7967
Iteration 2:  Log likelihood = -4476.3965
Iteration 3:  Log likelihood = -4476.3448
Iteration 4:  Log likelihood = -4476.3447
```

Two-parameter logistic model Number of obs = 1,500
Log likelihood = -4476.3447

Group: Male

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Discrim	1.187923	.1804778	6.58	0.000	.8341933	1.541653
Diff	-.5507796	.0894536	-6.16	0.000	-.7261054	-.3754538
q2						
Discrim	.90663	.1318739	6.87	0.000	.6481618	1.165098
Diff	-.0450698	.0761722	-0.59	0.554	-.1943645	.104225
q3						
Discrim	.8828704	.1462984	6.03	0.000	.5961307	1.16961
Diff	-1.703158	.2385734	-7.14	0.000	-2.170753	-1.235563
q4						
Discrim	.8196789	.1221824	6.71	0.000	.5802057	1.059152
Diff	.3770973	.0993197	3.80	0.000	.1824342	.5717603
q5						
Discrim	1.439933	.2218141	6.49	0.000	1.005185	1.874681
Diff	1.197739	.1437481	8.33	0.000	.9159978	1.47948
mean(Theta)	0	(omitted)				
var(Theta)	1	(constrained)				

Group: Female

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Discrim	1.187923	.1804778	6.58	0.000	.8341933	1.541653
Diff	-.5507796	.0894536	-6.16	0.000	-.7261054	-.3754538
q2						
Discrim	.90663	.1318739	6.87	0.000	.6481618	1.165098
Diff	-.0450698	.0761722	-0.59	0.554	-.1943645	.104225
q3						
Discrim	.8828704	.1462984	6.03	0.000	.5961307	1.16961
Diff	-1.703158	.2385734	-7.14	0.000	-2.170753	-1.235563
q4						
Discrim	.8196789	.1221824	6.71	0.000	.5802057	1.059152
Diff	.3770973	.0993197	3.80	0.000	.1824342	.5717603
q5						
Discrim	1.439933	.2218141	6.49	0.000	1.005185	1.874681
Diff	1.197739	.1437481	8.33	0.000	.9159978	1.47948
mean(Theta)	-.1348222	.0721434	-1.87	0.062	-.2762206	.0065763
var(Theta)	.6239155	.1239068			.4227474	.9208111

By default, the item parameters (the difficulty and discrimination) are constrained to be equal across groups. The mean and variance of the reference group (males) are constrained to 0 and 1, while the mean and variance of the focal group (females) are estimated.

We use `estat greport` to arrange the output in a more compact and readable format.

```
. estat greport
```

Parameter	Male	Female
q1		
Discrim	1.1879233	1.1879233
Diff	-.55077963	-.55077963
q2		
Discrim	.90662997	.90662997
Diff	-.04506976	-.04506976
q3		
Discrim	.88287037	.88287037
Diff	-1.7031579	-1.7031579
q4		
Discrim	.81967885	.81967885
Diff	.37709727	.37709727
q5		
Discrim	1.4399331	1.4399331
Diff	1.1977389	1.1977389
mean(Theta)	0	-.13482217
var(Theta)	1	.6239155

Now we can clearly see which parameters are constrained across groups and which are estimated freely.

We store our estimates for later use.

```
. estimates store nodif
```

◁

Differential item functioning

Differential item functioning occurs when respondents with the same ability have different probabilities of succeeding on a given item. This difference in probabilities may be caused by a shift in the discrimination parameter (a-DIF), the difficulty parameter (b-DIF), or both (ab-DIF). We begin with the most general case of ab-DIF.

▷ Example 2: Nonuniform DIF (ab-DIF)

Suppose we suspect item q4 behaves differently across groups and want to test for a shift in the a and b parameters. We let item q4 parameters vary across groups, and we keep the coefficients on the remaining items constrained to be the same across the groups. By specifying `(0: 2pl q4) (1: 2pl q4)`, we tell `irt` to estimate separate discrimination and difficulty parameters for item q4 for each group.

```
. irt (0: 2pl q4) (1: 2pl q4) (2pl q1 q2 q3 q5), group(female)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -4588.2619
Iteration 1:  Log likelihood = -4584.1505
Iteration 2:  Log likelihood =  -4584.149
Iteration 3:  Log likelihood =  -4584.149
Group: Male
Group: Female
Fitting full model:
Iteration 0:  Log likelihood = -4538.2523  (not concave)
Iteration 1:  Log likelihood = -4484.1177
Iteration 2:  Log likelihood = -4469.9992
Iteration 3:  Log likelihood = -4469.5296
Iteration 4:  Log likelihood = -4469.5261
Iteration 5:  Log likelihood = -4469.5261
Hybrid IRT model                                        Number of obs = 1,500
Log likelihood = -4469.5261
Group: Male
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q4						
Discrim	.6144966	.128936	4.77	0.000	.3617867	.8672065
Diff	.2554018	.1365998	1.87	0.062	-.0123289	.5231326
2pl						
q1						
Discrim	1.306988	.1969137	6.64	0.000	.9210443	1.692932
Diff	-.4777569	.0820373	-5.82	0.000	-.6385471	-.3169668
q2						
Discrim	.9316811	.1344814	6.93	0.000	.6681023	1.19526
Diff	-.0097118	.0749322	-0.13	0.897	-.1565762	.1371525
q3						
Discrim	.930189	.1493857	6.23	0.000	.6373984	1.22298
Diff	-1.587128	.2173969	-7.30	0.000	-2.013218	-1.161038
q5						
Discrim	1.445573	.2228226	6.49	0.000	1.008848	1.882297
Diff	1.208255	.1453832	8.31	0.000	.9233095	1.493201
mean(Theta)	0	(omitted)				
var(Theta)	1	(constrained)				

Group: Female

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q4						
Discrim	1.354599	.3649997	3.71	0.000	.6392133	2.069986
Diff	.3935696	.1292876	3.04	0.002	.1401707	.6469686
2pl						
q1						
Discrim	1.306988	.1969137	6.64	0.000	.9210443	1.692932
Diff	-.4777569	.0820373	-5.82	0.000	-.6385471	-.3169668
q2						
Discrim	.9316811	.1344814	6.93	0.000	.6681023	1.19526
Diff	-.0097118	.0749322	-0.13	0.897	-.1565762	.1371525
q3						
Discrim	.930189	.1493857	6.23	0.000	.6373984	1.22298
Diff	-1.587128	.2173969	-7.30	0.000	-2.013218	-1.161038
q5						
Discrim	1.445573	.2228226	6.49	0.000	1.008848	1.882297
Diff	1.208255	.1453832	8.31	0.000	.9233095	1.493201
mean(Theta)	-.0628546	.0706378	-0.89	0.374	-.201302	.0755929
var(Theta)	.5030102	.1159462			.3201628	.7902832

We use `estat greport` to arrange the output in a more compact and readable format.

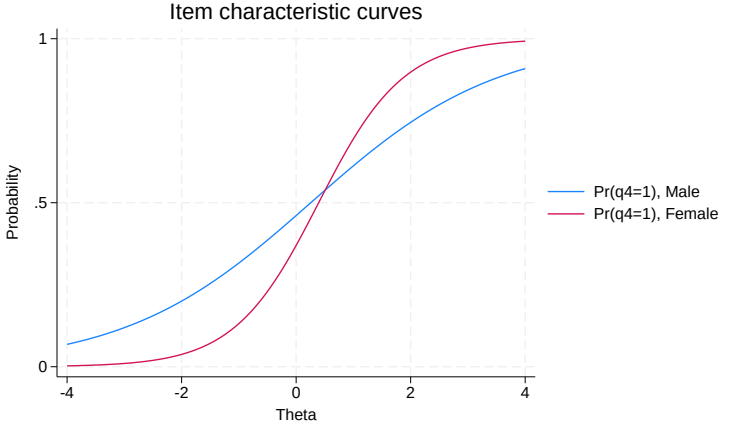
```
. estat greport
```

Parameter	Male	Female
q4		
Discrim	.6144966	1.3545995
Diff	.25540185	.39356964
q1		
Discrim	1.306988	1.306988
Diff	-.47775692	-.47775692
q2		
Discrim	.93168108	.93168108
Diff	-.00971184	-.00971184
q3		
Discrim	.93018903	.93018903
Diff	-1.5871277	-1.5871277
q5		
Discrim	1.4455726	1.4455726
Diff	1.2082554	1.2082554
mean(Theta)	0	-.06285455
var(Theta)	1	.50301021

Now it is easy to see that both the discrimination and difficulty parameters for item q4 differ between groups.

We plot the ICCs for item q4 for both groups using `irtgraph icc`; see [IRT] **irtgraph icc** for details.

```
. irtgraph icc q4
```



The graph suggests item q4 exhibits a significant amount of DIF. We confirm it formally using a likelihood-ratio test; see [R] **lrtest** for details.

```
. lrtest nodif .
Likelihood-ratio test
Assumption: nodif nested within .
 LR chi2(2) =  13.64
Prob > chi2 = 0.0011
```

We reject the null hypothesis that the discrimination and difficulty parameters are the same across the groups and conclude that item q4 exhibits a-DIF, b-DIF, or ab-DIF.

◁

▷ Example 3: Uniform DIF (b-DIF)

Suppose we wish to test whether a model that allows the difficulty of q4 to differ across groups fits better than a model where the difficulty is constrained.

We impose the required equality constraint on the discrimination, the a parameter, for item q4 using the cns() option. The constraints are on the discrimination, the a parameter. The @k is a symbolic constraint that tells irt that parameters adorned with the same symbol should be constrained to be equal; see [IRT] **irt constraints** for details.

```
. irt (0: 2pl q4, cns(a@k)) (1: 2pl q4, cns(a@k))
>     (2pl q1 q2 q3 q5), group(female)
```

(*output omitted*)

```
. estat greport
```

Parameter	Male	Female
q4		
Discrim	.78202733	.78202733
Diff	.20904803	.64854407
q1		
Discrim	1.2249413	1.2249413
Diff	-.50829857	-.50829857
q2		
Discrim	.91473677	.91473677
Diff	-.01287912	-.01287912
q3		
Discrim	.89618684	.89618684
Diff	-1.651863	-1.651863
q5		
Discrim	1.4209752	1.4209752
Diff	1.2393731	1.2393731
mean(Theta)	0	-.07076921
var(Theta)	1	.62679028

We see that in this model the estimated discrimination parameter for item q4 is the same between groups and the estimated difficulty parameter differs.

We compare this model with the fully constrained model from example 1 and conclude that the current model is preferable.

```
. lrtest nodif .
Likelihood-ratio test
Assumption: nodif nested within .
 LR chi2(1) =   8.21
Prob > chi2 = 0.0042
```

Reference

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Also see

[IRT] **irt, group( ) postestimation** — Postestimation tools for group IRT

[IRT] **DIF** — Introduction to differential item functioning

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

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

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

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

[U] 20 Estimation and postestimation commands

Postestimation commands predict Methods and formulas Also see

Postestimation commands

The following postestimation commands are of special interest after `irt`:

Command	Description
`estat greport`	report estimated group IRT parameters
`estat report`	report estimated IRT parameters
`irtgraph icc`	plot item characteristic curve (ICC)
`irtgraph iif`	plot item information function (IIF)
`irtgraph tcc`	plot test characteristic curve (TCC)
`irtgraph tif`	plot test information function (TIF)

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat (svy)`	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
* `lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	probabilities, linear predictions, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

*`lrtest` is not appropriate with `svy` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, and parameter-level scores.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of item probabilities and other statistics

`predict` [*type*] *newvarsspec* [*if*] [*in*] [, *statistic item_options*]

Syntax for obtaining estimated latent variables and their standard errors

`predict` [*type*] *newvarsspec* [*if*] [*in*], `latent` [*latent_options*]

Syntax for obtaining parameter-level scores

`predict` [*type*] *newvarsspec* [*if*] [*in*], `scores`

newvarsspec is *stub*`*` or *newvarlist*.

statistic	Description
Main	
`pr`	probabilities; the default
`xb`	linear prediction

item_options	Description
Main	
† `outcome(`*item* [*#*]`)`	specify item variable; default is all variables
`conditional(`*ctype*`)`	compute *statistic* conditional on estimated latent variables; default is `conditional(ebmeans)`
`marginal`	compute *statistic* marginally with respect to the latent variables
Integration	
int_options	integration options

† `outcome(`*item* *#*`)` may also be specified as `outcome(`*#.item*`)` or `outcome(`*item* `#`*#*`)`. `outcome(`*item* `#3)` means the third outcome value. `outcome(`*item* `#3)` would mean the same as `outcome(`*item* `4)` if outcomes were 1, 3, and 4.

ctype	Description
`ebmeans`	empirical Bayes means of latent variables; the default
`ebmodes`	empirical Bayes modes of latent variables
`fixedonly`	prediction for the fixed portion of the model only

latent_options	Description
Main	
ebmeans	use empirical Bayes means of latent trait; the default
ebmodes	use empirical Bayes modes of latent trait
se(*newvar*)	calculate standard errors
Integration	
int_options	integration options

int_options	Description
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 for predict

Main

pr, the default, calculates the predicted probability.

xb specifies that the linear predictor be calculated.

outcome(*item* [#]) specifies that predictions for *item* be calculated. Use # to specify which outcome level to predict. Predictions for all observed response variables are computed by default.

conditional(*ctype*) and marginal specify how latent variables are handled in computing *statistic*.

conditional() specifies that *statistic* will be computed conditional on specified or estimated 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, which means that *statistic* is calculated by integrating the prediction function with respect to all the latent variables over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates.

latent specifies that the latent trait is predicted using an empirical Bayes estimator; see options ebmeans and ebmodes.

ebmeans specifies that empirical Bayes means are used to predict the latent variables.

ebmodes specifies that empirical Bayes modes are used to predict the latent variables.

se(*newvar*) calculates standard errors of the empirical Bayes estimator and stores the result in *newvar*. This option requires the latent option.

scores calculates the scores for each coefficient in e(b). This option requires a new variable list of the length equal to the number of columns in e(b). Otherwise, use *stub*∗ to have predict generate enumerated variables with prefix *stub*.

Integration

intpoints(#) specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

iterate(#) specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

tolerance(#) specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

Methods and formulas

The predictions for multiple-group models are computed in the same way as predictions for single-group models. For binary items, see *Methods and formulas* of the postestimation entries for irt 1pl, irt 2pl, and irt 3pl. For ordered items, see *Methods and formulas* of the postestimation entries for irt grm, irt pcm, and irt rsm. For nominal items, see *Methods and formulas* of the postestimation entry for irt nrm. For hybrid models, see *Methods and formulas* of the postestimation entry for irt hybrid.

Also see

[IRT] **irt, group( )** — IRT models for multiple groups

[IRT] **estat greport** — Report estimated group IRT parameters

[IRT] **estat report** — Report estimated IRT parameters

[IRT] **irtgraph icc** — Item characteristic curve plot

[IRT] **irtgraph iif** — Item information function plot

[IRT] **irtgraph tcc** — Test characteristic curve plot

[IRT] **irtgraph tif** — Test information function plot

[U] 20 Estimation and postestimation commands

Description Quick start Syntax Remarks and examples Also see

Description

Constraints are imposed on the estimated parameters of a model. `irt` allows you to constrain a parameter to a fixed value or to constrain two or more parameters to be equal.

Quick start

2PL model for binary items `b1` to `b5`, with the discrimination parameters for items `b1` and `b2` constrained to be equal using symbolic constraints

```
irt ///
  (2pl b1, cns(a@k)) ///
  (2pl b2, cns(a@k)) ///
  (2pl b3-b5)
```

Same as above, but with the discrimination and difficulty parameters for items `b1` and `b2` constrained to fixed values

```
irt ///
  (2pl b1, cns(a@0.9 b@-1)) ///
  (2pl b2, cns(a@1.2 b@1.5)) ///
  (2pl b3-b5)
```

2PL model for binary items `b1` to `b5` and GRM for ordinal items `o1` to `o5` with discrimination parameters constrained to be equal for all items

```
irt ///
  (2pl b1-b5, cns(a@k)) ///
  (grm o1-o5, cns(a@k))
```

GRM for ordinal items `o1` to `o5` with the parameters of item `o1` constrained to fixed values

```
irt ///
  (grm o1, cns(a@1 b1@-1 b2@0 b3@1)) ///
  (grm o2-05)
```

1PL model for two groups with group-common items `b3-b7` and group-specific items `b1`, `b2`, `b8`, `b9` with the discrimination parameter constrained to be the same for all items

```
irt ///
  (0: 1pl b1 b2, cns(a@k)) ///
  (  1pl b3-b7, cns(a@k)) ///
  (1: 1pl b8 b9, cns(a@k)) ///
  , group(female)
```

Syntax

```
irt ... [ , cns(spec [spec ...]) ... ]
```

where *spec* is *parm*`@`# or *parm*`@`*symbol*.

In 1PL and 2PL models, *parm* is one of `a` or `b`, which corresponds to the discrimination or difficulty parameter in the IRT parameterization, or *parm* is one of `alpha` or `beta`, which corresponds to the slope or intercept in the slope-intercept parameterization.

In 3PL models, *parm* is one of `a`, `b`, or `c`, which corresponds to the discrimination, difficulty, or guessing parameter in the IRT parameterization, or *parm* is one of `alpha` or `beta`, which corresponds to the slope or intercept in the slope-intercept parameterization.

In nominal response models, *parm* is one of `a1`, `a2`, … for the multiple discrimination parameters per item, or *parm* is one of `b1`, `b2`, … for the multiple difficulty parameters per item.

In graded response, partial credit, and rating scale models, *parm* is `a` for the discrimination parameter, or *parm* is one of `b1`, `b2`, … for the multiple difficulty parameters per item.

`a` is a synonym for `a1`, and `b` is a synonym for `b1`.

Remarks and examples

Remarks are presented under the following headings:

Overview

Two types of constraints can be specified in IRT models:

1. Fixed-value constraints set a parameter to a specific value. These constraints are specified by using `@` and the value of the constraint, for example, `a@1.5`.
2. Symbolic constraints set parameters to be equal to each other. Symbolic constraints are specified by using `@` and a name, for example, `a@k1`. Symbolic names are just names from 1 to 32 characters in length.

Constraints in 1PL, 2PL, and 3PL models

IRT models can be written using the IRT or the slope-intercept parameterization. Constraints in 1PL, 2PL, and 3PL models are applied to a, b, and c of the IRT parameterization. For instance, in *Methods and formulas* in [IRT] **irt 2pl**, we show that the 2PL model can be written in the IRT parameterization as

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

where a_i represents discrimination of item i and b_i represents the difficulty of item i. For 2PL models, we can specify constraints on these a_i and b_i parameters. Similarly, constraints on 1PL models can be applied to a and b_i parameters. The 3PL model also allows constraints on the c parameter; see *Methods and formulas* in [IRT] **irt 3pl** for information on the IRT parameterization of this model.

The rules for specifying constraints in the IRT parameterization are the following:

1. Both fixed-value and symbolic constraints are allowed on the discrimination parameter, a.

   For instance, to constrain the discrimination parameter to 0.8 in a 1PL model, we could type

   ```
   . irt 1pl q1-q10, cns(a@0.8)
   ```

   To constrain all discrimination parameters in a 2PL model to be equal, reducing it to a 1PL model, we type

   ```
   . irt 2pl q1-q10, cns(a@k1)
   ```

2. Fixed-value constraints are allowed on the difficulty parameter b when a fixed-value constraint is also set on the corresponding a.

   For instance, in a 2PL model, to constrain the discrimination parameter to 0.8 and the difficulty parameter to -2 for item q1, we could type

   ```
   . irt                          ///
       (2pl q1, cns(a@0.8 b@-2)) ///
       (2pl q2-q10)
   ```

   Notice that we use the hybrid syntax to separate the item on which we are placing constraints from items that have no parameter constraints; see [IRT] **irt hybrid**.

3. Both fixed-value and symbolic constraints are allowed on the guessing parameter, c, in the 3PL model.

   To constrain the guessing parameter for all items to 0.1, we could type

   ```
   . irt 3pl q1-q10, cns(c@0.1)
   ```

   To constrain the guessing parameter for q1 and q2 to be equal (and different from the common guessing parameter for q3-q10), we could type

   ```
   . irt                   ///
       (3pl q1, cns(c@k1)) ///
       (3pl q2, cns(c@k1)) ///
       (3pl q3-q10)
   ```

We can also set constraints on the α and β parameters in the slope-intercept parameterization. For instance, in *Methods and formulas* in [IRT] **irt 2pl**, we show that the slope-intercept parameterization of the 2PL model is

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

where α_i is the slope for item i and β_i is the intercept for item i. 1PL and 3PL models have corresponding slope-intercept parameterizations.

Constraints on α and β are most often used in group IRT models to test for differential item functioning.

The rules for specifying constraints in the slope-intercept parameterization are the following:

1. Symbolic constraints are allowed on the slope parameter, α.

   To constrain the slope of item q1 to be equal across groups and let the intercepts vary across groups, we could type

```
. irt                           ///
    (0: 2pl q1, cns(alpha@k1)) ///
    (1: 2pl q1, cns(alpha@k1)) ///
    (2pl q2-q10)
```

2. Symbolic constraints are allowed on the intercept parameter, β.

   To constrain the intercept of item q1 to be equal across groups and let the slopes vary across groups, we could type

```
. irt                          ///
    (0: 2pl q1, cns(beta@k1)) ///
    (1: 2pl q1, cns(beta@k1)) ///
    (2pl q2-q10)
```

▷ Example 1: Fixed-value constraints in a 2PL model

Using masc1.dta, we could fit a 2PL model by typing

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt (2pl q1) (2pl q2 q3 q4)
```

Notice that we used a hybrid specification of our 2PL model with q1 separated from the other items. They hybrid syntax is commonly used when fitting models with constraints because it allows us to apply constraints to a chosen item or set of items; see [IRT] **irt hybrid** for information on this syntax.

We now constrain the discrimination to 1.5 and the difficulty to −0.5 for q1 by adding the cns(a@1.5 b@-0.5) option.

```
. irt (2pl q1, cns(a@1.5 b@-.5)) (2pl q2 q3 q4)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -2042.1777
Iteration 1:  Log likelihood =  -2041.492
Iteration 2:  Log likelihood = -2041.4917
Iteration 3:  Log likelihood = -2041.4917
Fitting full model:
Iteration 0:  Log likelihood = -2000.6715
Iteration 1:  Log likelihood = -1995.1431
Iteration 2:  Log likelihood = -1995.0644
Iteration 3:  Log likelihood = -1995.0643
Hybrid IRT model                                Number of obs = 800
Log likelihood = -1995.0643
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q1						
Discrim	1.5	(constrained)				
Diff	-.5	(constrained)				
2pl						
q2						
Discrim	.70149	.1469275	4.77	0.000	.4135175	.9894626
Diff	-.1455087	.1131457	-1.29	0.198	-.3672702	.0762528
q3						
Discrim	.9136072	.1960573	4.66	0.000	.5293419	1.297873
Diff	-1.726684	.2977765	-5.80	0.000	-2.310316	-1.143053
q4						
Discrim	.742854	.1556356	4.77	0.000	.4378138	1.047894
Diff	.3541431	.1220373	2.90	0.004	.1149543	.5933319

We see that the fixed-value constraints on q1 appear in the first table in the output.

◁

▷ Example 2: Constraining slopes in a group 2PL model

IRT models are reported using the IRT parameterization where $b_i = -\beta_i/\alpha_i$ and $a_i = \alpha_i$. Typically, constraints are set on the a's and b's as in the previous example. Note, however, that to set a constraint on one of the b's, we need to constrain both the underlying α and β parameters. Therefore, we are required to set a constraint on a any time we set a constraint on b.

Sometimes, we instead set constraints directly on the β parameters of the slope-intercept metric. Constraints on β do not require constraints on corresponding α parameters. Constraints specified in this slope-intercept metric are most often used in the context of multiple-group models.

For example, in a group 2PL model, we may want to constrain the intercepts to be the same across groups while allowing the slopes to differ across groups. Using `masc2.dta`, we can fit a group 2PL model by typing

```
. use https://www.stata-press.com/data/r19/masc2
(Data from De Boeck & Wilson (2004))
. irt (0: 2pl q1) (1: 2pl q1) (2pl q2 q3 q4), group(female)
```

This model allows the slope and intercept for `q1` to differ across groups and constrains the slopes and intercepts for all other items across groups; see [IRT] **irt, group( )** for information on group IRT models.

Now we can constrain the intercept for item `q1` to be equal across the two groups and allow only the slope of `q1` to vary across groups by typing

```
. irt (0: 2pl q1, cns(beta@k1)) (1: 2pl q1, cns(beta@k1)) (2pl q2 q3 q4),
> group(female)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -3848.9104
Iteration 1:  Log likelihood = -3845.4263
Iteration 2:  Log likelihood = -3845.4252
Iteration 3:  Log likelihood = -3845.4252
Group: Male
Group: Female
Fitting full model:
Iteration 0:  Log likelihood = -3786.3332
Iteration 1:  Log likelihood = -3779.6101
Iteration 2:  Log likelihood = -3778.4145
Iteration 3:  Log likelihood = -3778.3648
Iteration 4:  Log likelihood = -3778.3646
Hybrid IRT model                                        Number of obs = 1,500
Log likelihood = -3778.3646
Group: Male
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q1						
Discrim	1.756291	.6093519	2.88	0.004	.5619834	2.950599
Diff	-.3976132	.0983202	-4.04	0.000	-.5903172	-.2049092

2pl

q2						
	Discrim	.7065962	.160211	4.41	0.000	.3925883	1.020604
	Diff	-.0184657	.1007856	-0.18	0.855	-.2160018	.1790704
q3						
	Discrim	.7360524	.1741027	4.23	0.000	.3948173	1.077287
	Diff	-2.006872	.4360328	-4.60	0.000	-2.861481	-1.152264
q4						
	Discrim	.602399	.1225595	4.92	0.000	.3621868	.8426112
	Diff	.5431222	.1520137	3.57	0.000	.2451809	.8410635
mean(Theta)		0	(omitted)				
var(Theta)		1	(constrained)				

Group: Female

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
2pl						
q1						
Discrim	1.000017	.2965882	3.37	0.001	.4187151	1.581319
Diff	-.6983125	.2424141	-2.88	0.004	-1.173435	-.2231896
2pl						
q2						
Discrim	.7065962	.160211	4.41	0.000	.3925883	1.020604
Diff	-.0184657	.1007856	-0.18	0.855	-.2160018	.1790704
q3						
Discrim	.7360524	.1741027	4.23	0.000	.3948173	1.077287
Diff	-2.006872	.4360328	-4.60	0.000	-2.861481	-1.152264
q4						
Discrim	.602399	.1225595	4.92	0.000	.3621868	.8426112
Diff	.5431222	.1520137	3.57	0.000	.2451809	.8410635
mean(Theta)	-.0903098	.1215655	-0.74	0.458	-.3285738	.1479542
var(Theta)	1.295406	.4927367			.6146562	2.730106

We do not see the estimates of β in this output, but we can use the `estmetric` option to display the results in the slope-intercept parameterization.

```
. irt, estmetric
Hybrid IRT model                                        Number of obs = 1,500
Log likelihood = -3778.3646
 ( 1)  [q1]0bn.female - [q1]1.female = 0
 ( 2)  [q2]0bn.female - [q2]1.female = 0
 ( 3)  [q2]0bn.female#c.Theta - [q2]1.female#c.Theta = 0
 ( 4)  [q3]0bn.female - [q3]1.female = 0
 ( 5)  [q3]0bn.female#c.Theta - [q3]1.female#c.Theta = 0
 ( 6)  [q4]0bn.female - [q4]1.female = 0
 ( 7)  [q4]0bn.female#c.Theta - [q4]1.female#c.Theta = 0
```

```
 ( 8)  [/]mean(Theta)#0bn.female = 0
 ( 9)  [/]var(Theta)#0bn.female = 1
Group: Male                                          Number of obs = 761
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Theta	1.756291	.6093519	2.88	0.004	.5619834	2.950599
_cons	.6983246	.1370853	5.09	0.000	.4296424	.9670067
q2						
Theta	.7065962	.160211	4.41	0.000	.3925883	1.020604
_cons	.0130478	.0709493	0.18	0.854	-.1260102	.1521058
q3						
Theta	.7360524	.1741027	4.23	0.000	.3948173	1.077287
_cons	1.477163	.0933292	15.83	0.000	1.294241	1.660085
q4						
Theta	.602399	.1225595	4.92	0.000	.3621868	.8426112
_cons	-.3271763	.0668258	-4.90	0.000	-.4581524	-.1962002
mean(Theta)	0	(omitted)				
var(Theta)	1	(constrained)				

```
Group: Female                                        Number of obs = 739
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Theta	1.000017	.2965882	3.37	0.001	.4187151	1.581319
_cons	.6983246	.1370853	5.09	0.000	.4296424	.9670067
q2						
Theta	.7065962	.160211	4.41	0.000	.3925883	1.020604
_cons	.0130478	.0709493	0.18	0.854	-.1260102	.1521058
q3						
Theta	.7360524	.1741027	4.23	0.000	.3948173	1.077287
_cons	1.477163	.0933292	15.83	0.000	1.294241	1.660085
q4						
Theta	.602399	.1225595	4.92	0.000	.3621868	.8426112
_cons	-.3271763	.0668258	-4.90	0.000	-.4581524	-.1962002
mean(Theta)	-.0903098	.1215655	-0.74	0.458	-.3285738	.1479542
var(Theta)	1.295406	.4927367			.6146562	2.730106

Now we see that the equality constraints on the β parameters for `q1` were imposed across groups. The β's are the intercepts labeled `_cons` with a value of 0.698.

◁

Constraints in graded response models

Constraints for the GRM differ from the 1PL, 2PL, and 3PL models because each item in a GRM has multiple b parameters. In the `cns()` option, we refer to these parameters as `b1`, `b2`, `b3`, ...

The rules for constraints in the IRT parameterization of the GRM are the following:

1. Both fixed-value and symbolic constraints are allowed on the discrimination parameter, a.

   For instance, to constrain the discrimination parameters for all items to 0.8, we could type

   ```
   . irt grm q1-q5, cns(a@0.8)
   ```

   To constrain all discrimination parameters to be equal, we type

   ```
   . irt grm q1-q5, cns(a@k1)
   ```

2. Fixed-value constraints are allowed on the difficulty parameters, b_1, b_2, ..., when a fixed-value constraint is also set on the corresponding a.

   For instance, to constrain the discrimination parameter to 0.8 and the difficulty parameters to -2, -1, and 0 for item `q1`, we could type

   ```
   . irt                                    ///
       (grm q1, cns(a@0.8 b1@-2 b2@-1 b3@0)) ///
       (grm q2-q5)
   ```

Constraints in nominal response models

Constraints for the NRM differ from the GRM models because each item in an NRM has multiple a parameters in addition to multiple b parameters. In the `cns()` option, we refer to these a parameters as `a1`, `a2`, `a3`, ...

The rules for constraints in the IRT parameterization of the NRM are the following:

1. Both fixed-value and symbolic constraints are allowed on the discrimination parameters, a_1, a_2, ...

   For instance, to constrain the discrimination parameters for item `q1` to 1 and 1.1, we could type

   ```
   . irt                         ///
       (nrm q1, cns(a1@1 a2@1.1) ///
       (nrm q2-q5)
   ```

   To constrain discrimination parameters for `q1` and `q2` to be equal, we type

   ```
   . irt                            ///
       (nrm q1 q2, cns(a1@k1 a2@k2) ///
       (nrm q3-q5)
   ```

2. Fixed-value constraints are allowed on the difficulty parameters, b_1, b_2, ..., when a fixed-value constraint is also set on the corresponding discrimination parameters a_1, a_2, ...

   For instance, to constrain the discrimination parameters to 1 and 1.1 and constrain the difficulty parameters to -1 and 1 for item `q1`, we could type

   ```
   . irt                                    ///
       (nrm q1, cns(a1@1 a2@1.1 b1@-1 b2@1)) ///
       (nrm q2-q5)
   ```

Constraints in partial credit models

The PCM and GPCM have multiple difficulty parameters for each item. In the `cns()` option, we refer to these parameters as `b1`, `b2`, `b3`, ...

The rules for constraints in the IRT parameterization of the PCM and GPCM are the following:

1. Both fixed-value and symbolic constraints are allowed on the discrimination parameter, a.

   For instance, to constrain the discrimination parameter in a PCM to 0.8, we could type

   ```
   . irt pcm q1-q5, cns(a@0.8)
   ```

   To constrain discrimination parameters on `q1` and `q2` to be equal in a GPCM, we type

   ```
   . irt                       ///
       (gpcm q1 q2, cns(a@k1)) ///
       (gpcm q3-q5)
   ```

2. Fixed-value constraints are allowed on the difficulty parameters, b_1, b_2, ..., when a fixed-value constraint is also set on the corresponding a.

   For instance, to constrain the discrimination parameter to 0.8 and the difficulty parameters to -2, -1, and 0 for item `q1` in a GPCM, we could type

   ```
   . irt                                      ///
       (gpcm q1, cns(a@0.8 b1@-2 b2@-1 b3@0)) ///
       (gpcm q2-q5)
   ```

Constraints in rating scale models

The RSM has multiple difficulty parameters for each item. In the `cns()` option, we refer to these parameters as `b1`, `b2`, `b3`, ...

The rules for constraints in the IRT parameterization of the PCM are the following:

1. Both fixed-value and symbolic constraints are allowed on the discrimination parameter, a.

   For instance, to constrain the discrimination parameter to 0.8, we could type

   ```
   . irt rsm q1-q5, cns(a@0.8)
   ```

2. Fixed-value constraints are allowed on the difficulty parameters, b_1, b_2, ..., when a fixed-value constraint is also set on the corresponding a and on all b's for the item.

   For instance, to constrain the discrimination parameter to 0.8 and the difficulty parameters to -2, -1, and 0 for item `q1`, we could type

   ```
   . irt                                    ///
       (rsm q1, cns(a@0.8 b1@-2 b2@-1 b3@0)) ///
       (rsm q2-q5)
   ```

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt, group( )** — IRT models for multiple groups

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

estat report — Report estimated IRT parameters

Description

`estat report` displays the estimated IRT parameters. Estimates can be reorganized and sorted by parameter type.

Quick start

1PL model for binary items `b1` to `b10`

```
irt 1pl b1-b10
```

Report results grouped by parameter type

```
estat report, byparm
```

Same as above, and sort items by estimated difficulty

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

2PL model for binary items `b1` to `b20` and NRM for nominal items `n1` to `n10`

```
irt (2pl b1-b20) (nrm n1-n10)
```

Report results only for items `b15` and `n5`

```
estat report b15 n5
```

Report nominal item results grouped by parameter type

```
estat report n*, byparm
```

Menu

Statistics > IRT (item response theory)

Syntax

```
estat report [varlist] [ , options ]
```

`sort(`*p*`[ , descending])`	sort items by the estimated *p* parameters; *p* may be `a`, `b`, or `c`
`byparm`	arrange table rows by parameter rather than by item
Main	
`alabel(`*string*`)`	specify the `a` parameter label; the default is `Discrim`
`blabel(`*string*`)`	specify the `b` parameter label; the default is `Diff`
`clabel(`*string*`)`	specify the `c` parameter label; the default is `Guess`
`seqlabel`	label parameters in sequential order
`post`	post estimated IRT parameters and their VCE as estimation results
Reporting	
`level(`#`)`	set confidence level; default is `level(95)`
`verbose`	display estimation output in long form
display_options	control columns and column formats
`coeflegend`	display legend instead of statistics

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`coeflegend` does not appear in the dialog box.

Options

`sort(`*p*`[ , descending])` requests that items be sorted according to parameter *p*, where *p* is one of `a`, `b`, or `c`.

`sort(a)` specifies that items be sorted according to the estimated discrimination parameters.

`sort(b)` specifies that items be sorted according to the estimated difficulty parameters.

`sort(c)` specifies that items be sorted according to the estimated pseudoguessing parameters. It is relevant only for a 3PL model when option `sepguessing` is specified.

`descending` requests that the sorted items be reported in descending order. Sorted items are reported in ascending order by default.

`byparm` requests that the table rows be grouped by parameter rather than by item.

Main

`alabel(`*string*`)` labels the discrimination parameters with *string*. The default label is `Discrim`.

`blabel(`*string*`)` labels the difficulty parameters with *string*. The default label is `Diff`.

`clabel(`*string*`)` labels the pseudoguessing parameters with *string*. The default label is `Guess`. This option applies only to 3PL models.

`seqlabel` labels the estimated difficulty parameters within each categorical item sequentially, starting from 1. In NRM, `seqlabel` also labels the estimated discrimination parameters within each item sequentially, starting from 1. This option applies only to categorical models.

post causes estat report to behave like a Stata estimation (e-class) command. estat report posts the vector of estimated IRT parameters along with the corresponding variance–covariance matrix to e(), so that you can treat the estimated IRT parameters just as you would results from any other estimation command. For example, you could use test to perform simultaneous tests of hypotheses on the parameters, or you could use lincom to create linear combinations.

Reporting

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

verbose causes a separate discrimination, difficulty, and pseudoguessing parameter to be displayed for each item, even if the parameters are constrained to be the same across items. This option is implied when option post is specified.

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

The following option is available with estat report but is not shown in the dialog box:

coeflegend; see [R] **Estimation options**. This option is allowed only with the post option.

Remarks and examples

The following discussion is about how to use estat report with irt estimation results. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

▷ Example 1: Sorting binary items

We illustrate the features of estat report on the 2PL model we fit in example 1 of [IRT] **irt 2pl**. First, we refit the model.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. 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

For models with many items, it is often convenient to arrange the output according to highest or lowest difficulty (b) or discrimination (a). estat report makes sorting in a desired order easy. Below we specify option sort(b) to cause estat report to display the items in ascending order of the estimated difficulty parameter.

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

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

Here we add the byparm option to cause estat report to arrange the table rows by parameter type then by item difficulty.

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

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

Finally, we can tell estat report that we want to see parameter estimates for selected items only. Below we choose items q3, q5, and q8 and use the blabel() option to change the default label of the difficulty parameter from Diff to Location.

```
. estat report q3 q5 q8, sort(b) byparm blabel(Location)
Two-parameter logistic model                              Number of obs = 800
Log likelihood = -4118.4697
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim						
q8	1.399926	.233963	5.98	0.000	.9413668	1.858485
q3	.9245051	.1569806	5.89	0.000	.6168289	1.232181
q5	.8956621	.1535128	5.83	0.000	.5947825	1.196542
Location						
q8	-1.714416	.1925531	-8.90	0.000	-2.091814	-1.337019
q3	-1.70918	.242266	-7.05	0.000	-2.184012	-1.234347
q5	1.591164	.2325918	6.84	0.000	1.135293	2.047036

◁

Stored results

`estat report` stores the following in `r()`:

Scalars

`r(level)`	confidence level

Matrices

`r(table)`	matrix containing the parameter estimates with their standard errors, test statistics, *p*-values, and confidence intervals
`r(b)`	vector of estimated IRT parameters
`r(b_pclass)`	parameter class
`r(V)`	variance–covariance matrix of the estimated IRT parameters

`estat report` with the `post` option also stores the following in `e()`:

Macros

`e(properties)`	`b V`

Matrices

`e(b)`	vector of estimated IRT parameters
`e(V)`	variance–covariance matrix of the estimated IRT parameters

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

estat greport — Report estimated group IRT parameters

Description

`estat greport` displays the estimated group IRT parameters.

Quick start

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

```
irt 2pl b1-b10, group(female)
```

Report results in a compact format

```
estat greport
```

Report standard errors in addition to coefficients

```
estat greport, se
```

Report results grouped by parameter type

```
estat greport, byparm
```

Same as above, and sort items by estimated difficulty

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

Menu

Statistics > IRT (item response theory)

Syntax

```
estat greport [ , options ]
```

`sort(`*p*`[ , descending ])`	sort items by the estimated *p* parameters; *p* may be `a`, `b`, or `c`
`byparm`	arrange table rows by parameter rather than by item
Main	
`alabel(`*string*`)`	specify the `a` parameter label; the default is `Discrim`
`blabel(`*string*`)`	specify the `b` parameter label; the default is `Diff`
`clabel(`*string*`)`	specify the `c` parameter label; the default is `Guess`
`seqlabel`	label parameters in sequential order
`post`	post estimated IRT parameters and their VCE as estimation results
Reporting	
`b[ (`*%fmt*`) ]`	how to format coefficients, which are always reported
`se[ (`*%fmt*`) ]`	report standard errors and use optional format
`t[ (`*%fmt*`) ]`	report *t* or *z* statistics and use optional format
`p[ (`*%fmt*`) ]`	report *p*-values and use optional format
`parmwidth(`#`)`	use # characters to display variable and parameter names
`grwidth(`#`)`	use # characters to display group names and statistics
`style(oneline)`	display vertical line after variable names; the default
`style(columns)`	display vertical lines separating columns
`style(noline)`	suppress all vertical lines
`grlabel(`*string*`)`	column labels for groups
`title(`*string*`)`	title for table

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

Options

`sort(`*p*`[ , descending ])` requests that items be sorted according to parameter *p*, where *p* is one of `a`, `b`, or `c`.

`sort(a)` specifies that items be sorted according to the estimated discrimination parameters.

`sort(b)` specifies that items be sorted according to the estimated difficulty parameters.

`sort(c)` specifies that items be sorted according to the estimated pseudoguessing parameters. It is relevant only for a 3PL model when option `sepguessing` is specified.

`descending` requests that the sorted items be reported in descending order. Sorted items are reported in ascending order by default.

`byparm` requests that the table rows be grouped by parameter rather than by item.

Main

`alabel(`*string*`)` labels the discrimination parameters with *string*. The default label is `Discrim`.

`blabel(`*string*`)` labels the difficulty parameters with *string*. The default label is `Diff`.

`clabel(`*string*`)` labels the pseudoguessing parameters with *string*. The default label is `Guess`. This option applies only to 3PL models.

`seqlabel` labels the estimated difficulty parameters within each categorical item sequentially, starting from 1. In NRM, `seqlabel` also labels the estimated discrimination parameters within each item sequentially, starting from 1. This option applies only to categorical models.

`post` causes `estat greport` to behave like a Stata estimation (e-class) command. `estat greport` posts the vector of estimated IRT parameters along with the corresponding variance–covariance matrix to `e()`, so that you can treat the estimated IRT parameters just as you would results from any other estimation command. For example, you could use `test` to perform simultaneous tests of hypotheses on the parameters, or you could use `lincom` to create linear combinations.

Reporting

`b(%`*fmt*`)` specifies how the coefficients are to be displayed. You might specify `b(%9.2f)` to make decimal points line up. There is also a `b` option, which specifies that coefficients be displayed, but that is just included for consistency with the `se`, `t`, and `p` options. Coefficients are always displayed.

`se`, `t`, and `p` specify that standard errors, t or z statistics, and p-values be displayed. The default is not to display them. `se(%`*fmt*`)`, `t(%`*fmt*`)`, and `p(%`*fmt*`)` specify that each be displayed and specify the display format to be used.

`parmwidth(`#`)` specifies the number of character positions used to display the names of the variables and parameters. The default is `parmwidth(12)`.

`grwidth(`#`)` specifies the number of character positions used to display the names of the groups and statistics. The default is `grwidth(12)`.

`style(`*stylespec*`)` specifies the style of the coefficient table.

`style(oneline)` specifies that a vertical line be displayed after the variables but not between the groups. This is the default.

`style(columns)` specifies that vertical lines be displayed after each column.

`style(noline)` specifies that no vertical lines be displayed.

`grlabel(`*string*`)` specifies the labels for the group columns. The default is to use value labels of the group variable or, if the group variable has no value labels, a factor-variable indicator for each level of the group variable.

`title(`*string*`)` specifies the title to appear above the table.

Remarks and examples

The following discussion is about how to use `estat greport` with `irt` estimation results. If you are new to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

Example 1: Sorting binary items

We illustrate the features of estat greport on a group 2PL model we fit in example 1 of [IRT] **irt, group( )**. First, we refit the model.

```
. use https://www.stata-press.com/data/r19/masc2
(Data from De Boeck & Wilson (2004))
. irt 2pl q1-q5, group(female)
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -4594.5412
Iteration 1:  Log likelihood = -4590.4516
Iteration 2:  Log likelihood = -4590.4502
Iteration 3:  Log likelihood = -4590.4502
Group: Male
Group: Female
Fitting full model:
Iteration 0:  Log likelihood = -4503.5396  (not concave)
Iteration 1:  Log likelihood = -4479.7967
Iteration 2:  Log likelihood = -4476.3965
Iteration 3:  Log likelihood = -4476.3448
Iteration 4:  Log likelihood = -4476.3447
Two-parameter logistic model                    Number of obs = 1,500
Log likelihood = -4476.3447
Group: Male
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Discrim	1.187923	.1804778	6.58	0.000	.8341933	1.541653
Diff	-.5507796	.0894536	-6.16	0.000	-.7261054	-.3754538
q2						
Discrim	.90663	.1318739	6.87	0.000	.6481618	1.165098
Diff	-.0450698	.0761722	-0.59	0.554	-.1943645	.104225
q3						
Discrim	.8828704	.1462984	6.03	0.000	.5961307	1.16961
Diff	-1.703158	.2385734	-7.14	0.000	-2.170753	-1.235563
q4						
Discrim	.8196789	.1221824	6.71	0.000	.5802057	1.059152
Diff	.3770973	.0993197	3.80	0.000	.1824342	.5717603
q5						
Discrim	1.439933	.2218141	6.49	0.000	1.005185	1.874681
Diff	1.197739	.1437481	8.33	0.000	.9159978	1.47948
mean(Theta)	0	(omitted)				
var(Theta)	1	(constrained)				

```
Group: Female
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
q1						
Discrim	1.187923	.1804778	6.58	0.000	.8341933	1.541653
Diff	-.5507796	.0894536	-6.16	0.000	-.7261054	-.3754538
q2						
Discrim	.90663	.1318739	6.87	0.000	.6481618	1.165098
Diff	-.0450698	.0761722	-0.59	0.554	-.1943645	.104225
q3						
Discrim	.8828704	.1462984	6.03	0.000	.5961307	1.16961
Diff	-1.703158	.2385734	-7.14	0.000	-2.170753	-1.235563
q4						
Discrim	.8196789	.1221824	6.71	0.000	.5802057	1.059152
Diff	.3770973	.0993197	3.80	0.000	.1824342	.5717603
q5						
Discrim	1.439933	.2218141	6.49	0.000	1.005185	1.874681
Diff	1.197739	.1437481	8.33	0.000	.9159978	1.47948
mean(Theta)	-.1348222	.0721434	-1.87	0.062	-.2762206	.0065763
var(Theta)	.6239155	.1239068			.4227474	.9208111

Group IRT models always produce a lot of output. The estimated parameters are reported separately for each group, and it is hard to visually compare estimates across groups. `estat greport` makes such comparisons a breeze.

```
. estat greport
```

Parameter	Male	Female
q1		
Discrim	1.1879233	1.1879233
Diff	-.55077963	-.55077963
q2		
Discrim	.90662997	.90662997
Diff	-.04506976	-.04506976
q3		
Discrim	.88287037	.88287037
Diff	-1.7031579	-1.7031579
q4		
Discrim	.81967885	.81967885
Diff	.37709727	.37709727
q5		
Discrim	1.4399331	1.4399331
Diff	1.1977389	1.1977389
mean(Theta)	0	-.13482217
var(Theta)	1	.6239155

For models with many items, it is often convenient to arrange the output according to highest or lowest difficulty (b) or discrimination (a). `estat greport` makes sorting in a desired order easy. Below, we specify option `sort(b)` to cause `estat greport` to display the items in ascending order of the estimated difficulty parameter. The sort is performed on the first group.

```
. estat greport, sort(b)
```

Parameter	Male	Female
q3		
Discrim	.88287037	.88287037
Diff	-1.7031579	-1.7031579
q1		
Discrim	1.1879233	1.1879233
Diff	-.55077963	-.55077963
q2		
Discrim	.90662997	.90662997
Diff	-.04506976	-.04506976
q4		
Discrim	.81967885	.81967885
Diff	.37709727	.37709727
q5		
Discrim	1.4399331	1.4399331
Diff	1.1977389	1.1977389
mean(Theta)	0	-.13482217
var(Theta)	1	.6239155

Finally, we add the `byparm` option to cause `estat greport` to arrange the table rows by parameter type and then by item difficulty.

```
. estat greport, sort(b) byparm
```

Parameter	Male	Female
Discrim		
q3	.88287037	.88287037
q1	1.1879233	1.1879233
q2	.90662997	.90662997
q4	.81967885	.81967885
q5	1.4399331	1.4399331
Diff		
q3	-1.7031579	-1.7031579
q1	-.55077963	-.55077963
q2	-.04506976	-.04506976
q4	.37709727	.37709727
q5	1.1977389	1.1977389
mean(Theta)	0	-.13482217
var(Theta)	1	.6239155

Stored results

estat greport stores the following in r():

Macros
- r(names) — labels used for group names

Matrices
- r(b) — vector of estimated IRT parameters
- r(b_pclass) — parameter class
- r(V) — variance–covariance matrix of the estimated IRT parameters

estat greport with the post option also stores the following in e():

Macros
- e(properties) — b V

Matrices
- e(b) — vector of estimated IRT parameters
- e(V) — variance–covariance matrix of the estimated IRT parameters

Also see

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

[IRT] **irt, group( )** — IRT models for multiple groups

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

irtgraph icc — Item characteristic curve plot

Description

`irtgraph icc` plots item characteristic curves (ICCs) for binary items and category characteristic curves (CCCs) for categorical items for the currently fitted IRT model.

Quick start

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

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

Plot ICCs for all items

```
irtgraph icc
```

Plot ICCs and item difficulties for items `b1`, `b5`, and `b9`

```
irtgraph icc b1 b5 b9, blocation
```

GRM for ordinal items `o1` to `o5`, items coded `1`, `2`, `3`

```
irt grm o1-o5
```

Plot CCCs for selected item categories

```
irtgraph icc 1.o1 3.o5 2.o1
```

Plot CCCs for the first category of all items

```
irtgraph icc 1.o*
```

Fit a group 2PL model

```
irt 2pl b1-b9, group(female)
```

Plot ICCs for selected items and groups

```
irtgraph icc (b1) (0: b5) (1: b9)
```

Menu

Statistics > IRT (item response theory)

Syntax

Basic syntax

```
irtgraph icc [varlist] [, options]
```

Full syntax

```
irtgraph icc ([#:] varlist [, plot_options]) ([#:] varlist [, plot_options]) [...]
    [, options]
```

varlist is a list of items from the currently fitted IRT model.

#: plots curves for the specified group; allowed only after a group IRT model.

options	Description
Plots	
`blocation[(`*line_options*`)]`	add vertical lines for estimated item difficulties
`plocation[(`*line_options*`)]`	add horizontal lines for midpoint probabilities
`bcc`	plot boundary characteristic curves for categorical items
`ccc`	plot category characteristic curves
`range(# #)`	plot over $\theta = \#$ to #
Line	
line_options	affect rendition of the plotted curves
Add plots	
`addplot(`*plot*`)`	add other plots to the ICC plot
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than `by()` documented in [G-3] ***twoway_options***
Data	
`n(#)`	evaluate curves at # points; default is `n(300)`
`data(`*filename*`[, replace])`	save plot data to a file

plot_options	Description
`blocation[(`*line_options*`)]`	add vertical lines for estimated item difficulties
`plocation[(`*line_options*`)]`	add horizontal lines for midpoint probabilities
`bcc`	plot boundary characteristic curves for categorical items
`ccc`	plot category characteristic curves
line_options	affect rendition of the plotted curves

varlist may use factor-variable notation; see **[U] 11.4.3 Factor variables**.

line_options in *plot_options* override the same options specified in *options*.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

Options

Plots

`blocation`[`(`*line_options*`)`] specifies that for each ICC, a vertical line be drawn from the estimated difficulty parameter on the x axis to the curve. The optional *line_options* specify how the vertical lines are rendered; see [G-3] ***line_options***. This option implies option `bcc`.

`plocation`[`(`*line_options*`)`] specifies that for each ICC, a horizontal line be drawn from the midpoint probability on the y axis to the curve. The optional *line_options* specify how the horizontal lines are rendered; see [G-3] ***line_options***. This option implies option `bcc`.

`bcc` specifies that boundary characteristic curves (BCCs) be plotted for categorical items. The ICCs for the individual item categories are plotted by default. This option has no effect on binary items.

`ccc` specifies that category characteristic curves (CCCs) be plotted for all items. This is the default behavior for categorical items. For binary items, this option will plot ICCs for both outcomes.

`range(# #)` specifies the range of values for θ. This option requires a pair of numbers identifying the minimum and maximum. The default is `range(-4 4)` unless the estimated difficulty parameters exceed these values, in which case the range is extended.

Line

line_options affect the rendition of the plotted ICCs; see [G-3] ***line_options***.

Add plots

`addplot(`*plot*`)` allows adding more `graph twoway` plots to the graph; see [G-3] ***addplot_option***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***).

Data

`n(#)` specifies the number of points at which the ICCs, CCCs, and BCCs are to be evaluated. The default is `n(300)`.

`data(`*filename* [`, replace`]`)` saves the plot data to a Stata data file.

Remarks and examples

`irtgraph icc` plots ICCs for binary items and CCCs for categorical items after estimating the parameters of an IRT model using `irt`.

ICCs are also known as item response functions and item response curves.

CCCs are also known as category response functions, option response functions, operating characteristic curves, and category response curves.

For categorical items, `irtgraph icc` also plots BCCs, which are probability curves for crossing a boundary. BCCs are also known as "category boundary curves".

`irtgraph icc` is very flexible, and the best way to learn its capabilities is through examples.

Example 1: ICCs for binary outcomes

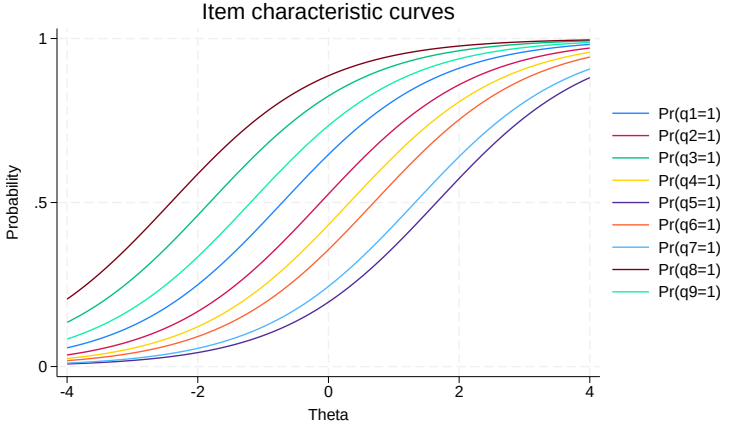
We continue with the model from example 1 of [IRT] **irt 1pl**. Recall that we fit a 1PL model to the nine binary items. Here we use `estat report` to rearrange the estimated IRT parameters sorted by item difficulty.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt 1pl q1-q9
  (output omitted)
. estat report, sort(b) byparm
One-parameter logistic model                    Number of obs = 800
Log likelihood = -4142.3516
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
Discrim	.852123	.0458445	18.59	0.000	.7622695	.9419765
Diff						
q8	-2.413443	.1691832	-14.27	0.000	-2.745036	-2.08185
q3	-1.817693	.1399523	-12.99	0.000	-2.091994	-1.543391
q9	-1.193206	.1162054	-10.27	0.000	-1.420965	-.965448
q1	-.7071339	.1034574	-6.84	0.000	-.9099066	-.5043612
q2	-.1222008	.0963349	-1.27	0.205	-.3110138	.0666122
q4	.3209596	.0976599	3.29	0.001	.1295498	.5123695
q6	.6930617	.1031842	6.72	0.000	.4908243	.8952991
q7	1.325001	.1205805	10.99	0.000	1.088668	1.561335
q5	1.652719	.1329494	12.43	0.000	1.392144	1.913295

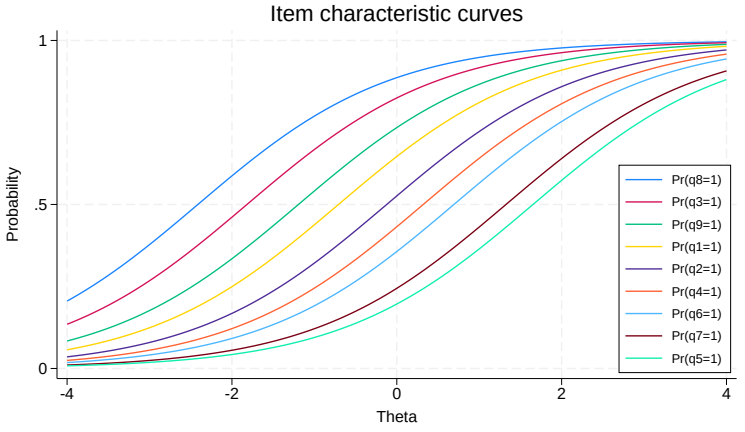
To plot ICCs for all items in the model, we simply type

```
. irtgraph icc
```



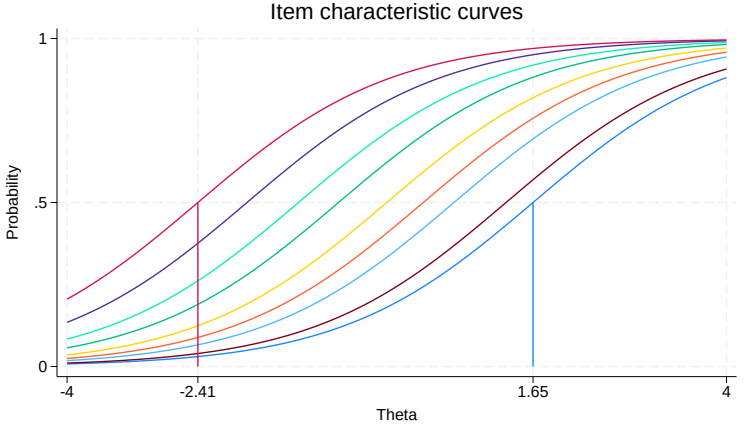
We can modify the default legend with the `legend()` option. Here, we shrink the legend and move it inside the plot region. We also specify a list of items explicitly so that the legend lists the ICCs in the order they appear in the graph. At the end of our interactive session, we came up with the following.

```
. irtgraph icc q8 q3 q9 q1 q2 q4 q6 q7 q5,
> legend(pos(4) ring(0) size(small) region(lcolor(black)))
```



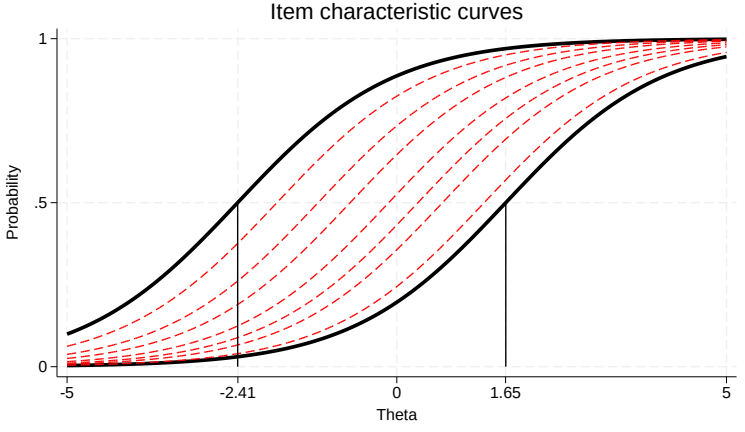
Another way to present the ICCs is to turn off the legend and highlight the items with the lowest and highest estimated difficulty parameter. From the output of `estat report`, we see that item `q8` has the lowest estimated difficulty parameter and item `q5` has the highest estimated difficulty parameter. Here we put those items in a separate plotting group, request their estimated difficulty locations be plotted, and put the remaining items in another plotting group.

```
. irtgraph icc (q5 q8, blocation) (q1-q4 q6 q7 q9), legend(off)
```



This plot shows the information we wanted, but we can tinker further to make items q5 and q8 stand out and make the whole plot more dramatic. Here is the final result.

```
. irtgraph icc
> (q8 q5, lcolor(black) lwidth(thick) bloc(lcolor(black)))
> (q1-q4 q6 q7 q9, lpattern(dash)),
> range(-5 5) xlabel(-5 -2.41 0 1.65 5) legend(off) lcolor(red)
```



We admit the above works nicely for a 1PL model because the ICCs do not cross; 2PL and 3PL models may require a different approach, but the general idea remains the same—we rarely obtain the desired ICC plot on the first try and need to work incrementally to arrive at the graph that best suits the estimated model parameters.

◁

❑ Technical note

For a binary item, it is standard practice to plot only the ICC for the probability of the positive outcome. Thus the following commands are equivalent.

```
. irtgraph icc q1
. irtgraph icc 1.q1
```

However, there are in fact two ICCs we could plot: one for the probability of the positive outcome and one for the probability of the negative outcome. To plot both ICCs, we can use any of the following:

```
. irtgraph icc 0.q1 1.q1
. irtgraph icc i.q1
. irtgraph icc q1, ccc
```

Because the two probabilities sum to 1, the ICC for the negative outcome is a mirror image of the ICC for the positive outcome reflected about the y axis at 0.5.

❑

Example 2: CCCs for categorical outcomes

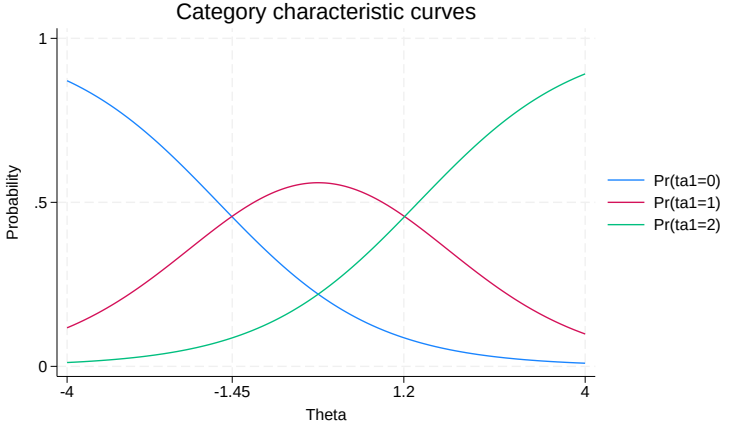
We continue with the model introduced in example 1 of [IRT] **irt grm**. To easily present some graphical features, we collapse the last two categories into one for all items and refit the GRM.

```
. use https://www.stata-press.com/data/r19/charity
(Data from Zheng & Rabe-Hesketh (2007))
. recode ta1-ta5 (3=2)
(58 changes made to ta1)
(102 changes made to ta2)
(55 changes made to ta3)
(36 changes made to ta4)
(86 changes made to ta5)
. irt grm ta1-ta5
```

(*output omitted*)

For a model with many categorical items, we do not recommend using `irtgraph icc` without *varlist*, because the resulting graph will contain far too many plotted curves. With 5 items, each with 3 categories, the total number of CCCs in the default plot is 15. Here we focus on item `ta1`.

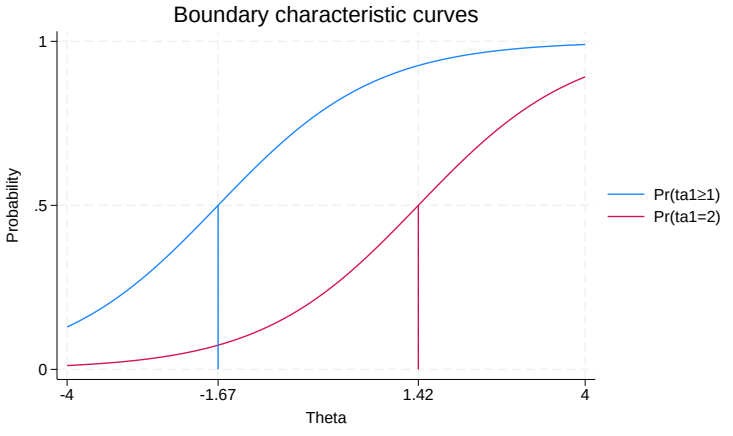
```
. irtgraph icc ta1, xlabel(-4 -1.45 1.20 4)
```



In a GRM, the adjacent probability curves do not cross at the estimated difficulty parameters. Each crossing point represents the level of the latent trait at which an examinee "transitions" from responding in one category versus the next. Thus, in the graph above, respondents whose trait level is below approximately -1.45 are most likely to answer 0 (strongly agree); respondents whose trait level is between approximately -1.45 and 1.20 are most likely to answer 1 (somewhat agree); and respondents whose trait level is above approximately 1.20 are most likely to answer 2 (somewhat or strongly disagree).

Because the GRM is defined in terms of cumulative probabilities, the estimated difficulties represent a point at which a person with $\theta = b_{ik}$ has a 50% chance of responding in category k or higher. We can use `irtgraph` to plot these probabilities with the corresponding estimated category difficulties. These probability curves are known as BCCs. We specify option `blocation`, which plots the category difficulties and also implies option `bcc`.

```
. irtgraph icc ta1, blocation
```



◁

❑ Technical note

In the example above, we typed

```
. irtgraph icc ta1
```

to plot the CCCs for item `ta1`. Because item `ta1` is coded 0, 1, or 2, we could have typed

```
. irtgraph icc 0.ta1 1.ta1 2.ta1
```

or

```
. irtgraph icc i.ta1
```

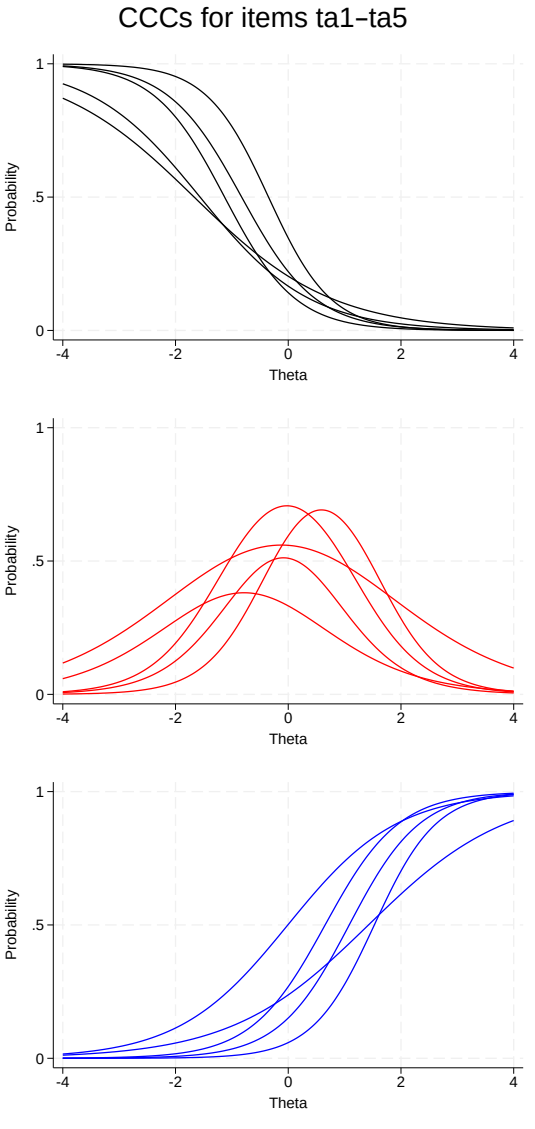
However, the first notation is most convenient to type. The factor notation comes in handy when we want to plot a particular category or change its appearance in the graph.

❑

▷ Example 3: Combining graphs

Sometimes, it is useful to focus on one category and plot its CCCs for all items. Below we show one way of presenting this information. We do not label the curves, because we want to see only the overall shape and location of the CCCs. We could always play with the legend to identify the individual curves, as we did above in example 1.

```
. irtgraph icc 0.ta*, legend(off) title("") lcolor(black) nodraw
> name(out0,replace)
. irtgraph icc 1.ta*, legend(off) title("") lcolor(red) nodraw
> name(out1,replace)
. irtgraph icc 2.ta*, legend(off) title("") lcolor(blue) nodraw
> name(out2,replace)
. graph combine out0 out1 out2, col(1) xsize(3) ysize(6)
> title("CCCs for items ta1-ta5")
```



▷ Example 4: ICCs for group IRT models

Here we demonstrate the behavior of `irtgraph icc` with group IRT models. The same comments apply to [IRT] **irtgraph iif**. If you are not familiar with group IRT modeling, we suggest you read [IRT] **irt, group( )** first.

We use an abridged version of the mathematics and science data from De Boeck and Wilson (2004). We fit the following model.

```
. use https://www.stata-press.com/data/r19/masc2, clear
(Data from De Boeck & Wilson (2004))
. irt (0: 1pl q1) (1: 1pl q1) (1pl q2 q3) (1: 1pl q4 q5) (0: 1pl q6),
> group(female)
```

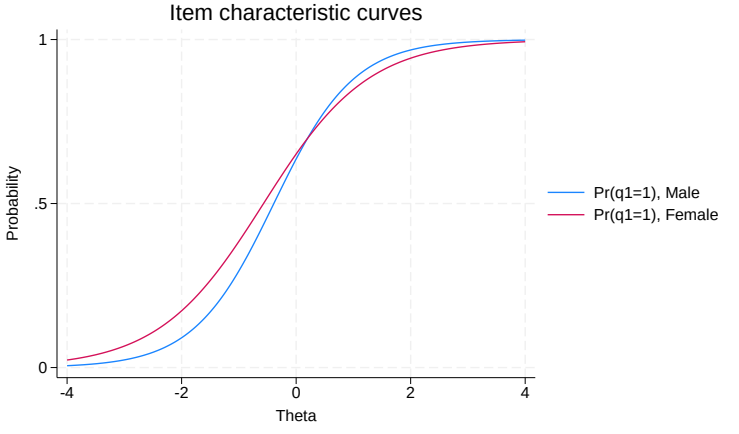
(*output omitted*)

```
. estat greport
```

Parameter	Male	Female
q1		
Discrim	1.430038	1.093494
Diff	-.38922567	-.56959664
q2		
Discrim	.82849958	.82849958
Diff	.04043785	.04043785
q3		
Discrim	.82849958	.82849958
Diff	-1.743068	-1.743068
q6		
Discrim	.99394585	
Diff	.73061294	
q4		
Discrim		.87968755
Diff		.7142615
q5		
Discrim		.87968755
Diff		2.2134151
mean(Theta)	0	.02971286
var(Theta)	1	.98927041

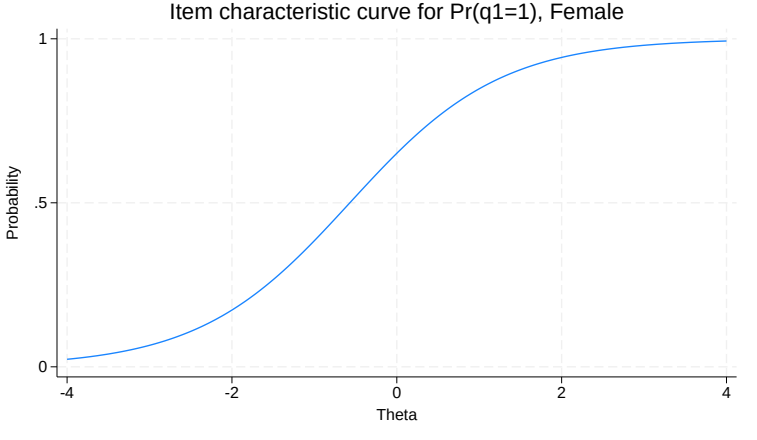
Looking at the output of `estat greport`, we see that both groups were administered item `q1` and that the parameters for this item differ between groups. In this situation, specifying `irtgraph icc q1` defaults to drawing the curves for both groups.

```
. irtgraph icc q1
```



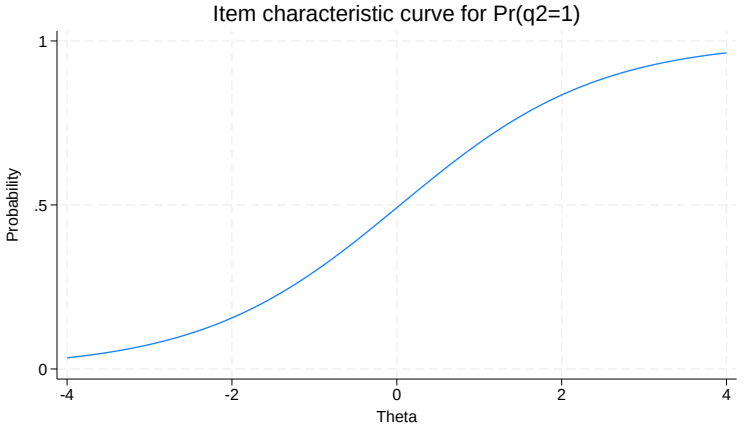
If we wish to restrict `irtgraph icc` to drawing a curve for a specific group, we use a group identifier before the item. Here we draw the ICC for item `q1` for the females only.

```
. irtgraph icc 1:q1
```



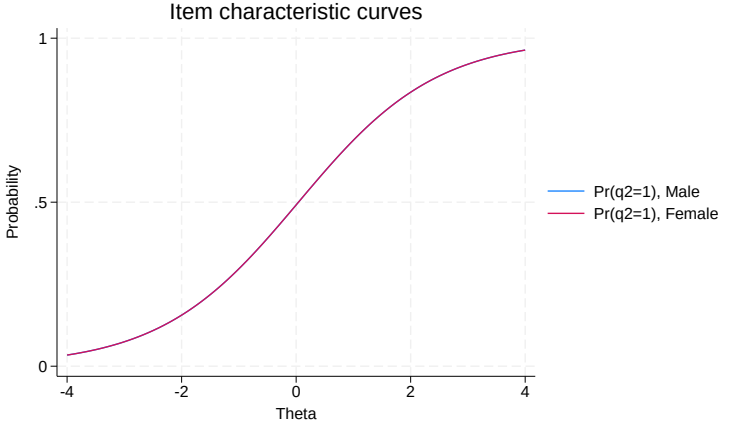
When item parameters are equal between groups, by default, `irtgraph icc` draws one ICC because the curve is the same for all groups. Here we graph the ICC for item q2.

```
. irtgraph icc q2
```



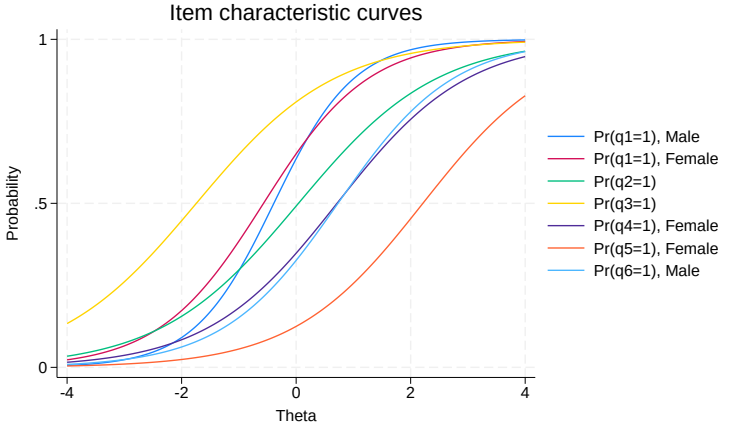
To convince yourself that the curves for item q2 are the same between groups, you can force `irtgraph icc` to draw separate curves for each group by putting each group in a separate equation.

```
. irtgraph icc (0:q2) (1:q2)
```



Now we can look at the ICC plot for all items.

```
. irtgraph icc
```



`q1` is shown twice because the parameters are different between groups. `q2` and `q3` are each shown once because the parameters are group invariant. `q4` and `q5` were administered only to the female group, so they are each shown only for the female group. `q6` was administered only to the male group, so it is shown only for the male group.

◁

Stored results

`irtgraph icc` stores the following in `r()`:

Macros

`r(xvals)`	values used to label the x axis
`r(yvals)`	values used to label the y axis

References

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Raciborski, R. 2015. Spotlight on irt. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2015/07/31/spotlight-on-irt/.

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

[IRT] **irtgraph tcc** — Test characteristic curve plot

irtgraph tcc — Test characteristic curve plot

Description

`irtgraph tcc` plots the test characteristic curve (TCC) for the currently fitted IRT model.

Quick start

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

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

Plot the TCC for the fitted model

```
irtgraph tcc
```

Plot the TCC, and show the expected score for the latent trait level of −1 and 1

```
irtgraph tcc, thetalines(-1 1)
```

Plot the TCC, and show the latent trait level for the expected scores of 5 and 8

```
irtgraph tcc, scorelines(5 8)
```

Fit a group 2PL model

```
irt 2pl b1-b9, group(female)
```

Plot the TCCs for both groups

```
irtgraph tcc
```

Plot the TCCs, and show the latent trait level for the expected scores of 5 and 8 for group 1

```
irtgraph tcc, scorelines(1: 5 8)
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irtgraph tcc [ , options ]
```

options	Description
Plots	
scorelines([#:]numlist [, refopts])	add x and y reference lines at each score value in *numlist*
thetalines([#:]numlist [, refopts])	add x and y reference lines at each θ value in *numlist*
range(# #)	plot over $\theta = \#$ to $\#$
Add plots	
addplot(*plot*)	add other plots to the TCC plot
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than by() documented in [G-3] ***twoway_options***
Data	
n(#)	evaluate curves at # points; default is n(300)
data(*filename* [, replace])	save plot data to a file

scorelines() and thetalines() can be specified multiple times.

#: plots lines for the specified group; allowed only after a group IRT model.

refopts	Description
line_options	affect rendition of the plotted expected score and θ lines
noxlines	suppress the corresponding reference lines for θ values
noylines	suppress the corresponding reference lines for score values

Options

Plots

scorelines([#:]*numlist* [, *refopts*]) adds x and y reference lines at each score value in *numlist*.

For group IRT models, reference lines are plotted for all groups. You can specify the optional #: to restrict reference lines to a specific group.

thetalines([#:]*numlist* [, *refopts*]) adds x and y reference lines at each θ value in *numlist*.

For group IRT models, reference lines are plotted for all groups. You can specify the optional #: to restrict reference lines to a specific group.

refopts affect the rendering of expected score and θ lines:

line_options specify how the expected score and θ lines are rendered; see [G-3] ***line_options***.

noxlines suppresses the corresponding reference line for θ.

noylines suppresses the corresponding reference line for scores.

range(# #) specifies the range of values for θ. This option requires a pair of numbers identifying the minimum and maximum. The default is range(-4 4).

Add plots

`addplot(`*plot*`)` allows adding more `graph twoway` plots to the graph; see [G-3] ***addplot_option***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***).

Data

`n(`#`)` specifies the number of points at which the TCCs are to be evaluated. The default is `n(300)`.

`data(`*filename* [`, replace`]`)` saves the plot data to a Stata data file.

Remarks and examples

`irtgraph tcc` plots the TCC after estimating the parameters of an IRT model using `irt`. The curve is also known as the "total characteristic curve". The TCC is the sum of ICCs for the entire instrument and thus plots the expected score on the test along the latent trait continuum.

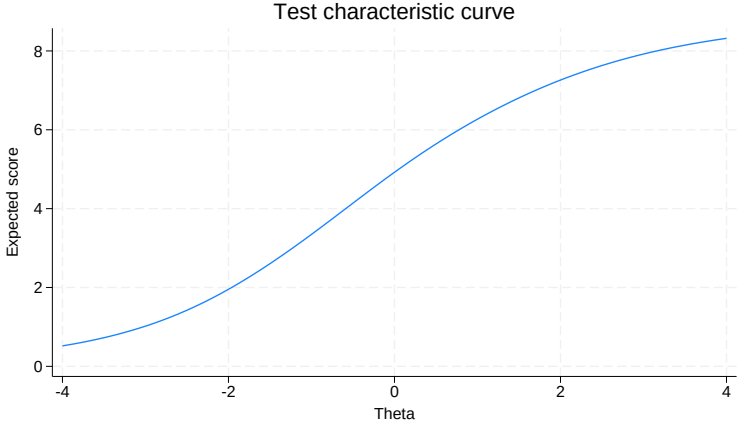
▷ Example 1: TCC for binary items

We continue with the 2PL model from example 1 of [IRT] **irt 2pl**. Recall that we fit a 2PL model to the nine binary items.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt 2pl q1-q9
  (output omitted)
```

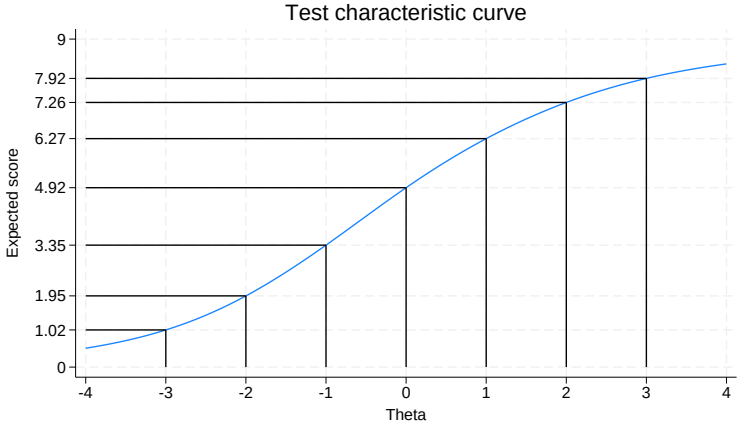
Because our instrument contains nine binary items, each coded as a 0 or a 1, the lowest possible score is 0, and the highest possible score is 9. Here we plot the TCC for the fitted model to see the expected score for a given range of the latent trait θ.

```
. irtgraph tcc
```



We can provide `irtgraph tcc` with a list of θ values, and `irtgraph tcc` will plot the corresponding expected scores. Here we ask for the expected scores corresponding to the latent trait level -3 to 3 in steps of 1.

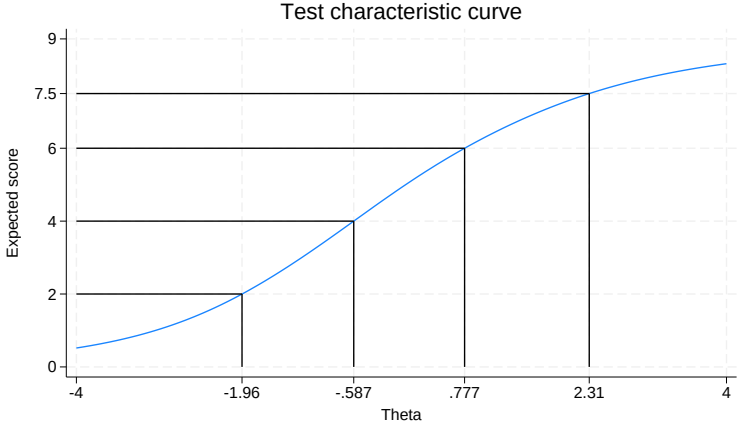
```
. irtgraph tcc, thetalines(-3/3)
```



We see that for an examinee with the latent trait level equal to 0, the expected score on the test is 4.92.

We can also provide `irtgraph tcc` with a list of expected scores, and `irtgraph tcc` will plot the corresponding latent trait values.

```
. irtgraph tcc, scorelines(2 4 6 7.5)
```



We see that examinees with a latent trait level of 0.78 and above are expected to obtain a score of 6 or more on the test.

◁

❑ Technical note

For nominal items, a score has no meaning other than to designate the response category, and a total score cannot be obtained. `irtgraph tcc` will plot the TCC in the presence of nominal outcomes, treating them as ordinal, and print a warning note alerting the user to the situation.

❑

Stored results

irtgraph tcc stores the following in r():

Macros

r(xvals)	values used to label the x axis
r(yvals)	values used to label the y axis

Reference

Raciborski, R. 2015. Spotlight on irt. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2015/07/31/spotlight-on-irt/.

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

[IRT] **irtgraph icc** — Item characteristic curve plot

irtgraph iif — Item information function plot

Description

`irtgraph iif` plots item information functions (IIFs) for items in the currently fitted IRT model.

Quick start

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

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

Plot IIFs for all items in the model

```
irtgraph iif
```

Plot IIFs for items `b1` and `b5`

```
irtgraph iif b1 b5
```

Fit a group 2PL model

```
irt 2pl b1-b9, group(female)
```

Plot IIFs for items `b1` and `b5` for both groups

```
irtgraph iif b1 b5
```

Plot IIFs for item `b1` for both groups and for item `b5` for group 1

```
irtgraph iif (b1) (1: b5)
```

Menu

Statistics > IRT (item response theory)

Syntax

Basic syntax

```
irtgraph iif [varlist] [, options]
```

Full syntax

```
irtgraph iif ([#:] varlist [, line_options]) ([#:] varlist [, line_options]) [...]
    [, options]
```

varlist is a list of items from the currently fitted IRT model.

#: selects items in *varlist* for the specified group.

options	Description
Plots	
`range(# #)`	plot over $\theta = \#$ to $\#$
Line	
line_options	affect rendition of the plotted IIFs
Add plots	
`addplot(`*plot*`)`	add other plots to the IIF plot
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than `by()` documented in [G-3] ***twoway_options***
Data	
`n(#)`	evaluate IIFs at # points; default is `n(300)`
`data(`*filename*`[, replace])`	save plot data to a file

line_options in (*varlist*, *line_options*) override the same options specified in *options*.

Options

Plots

`range(# #)` specifies the range of values for θ. This option requires a pair of numbers identifying the minimum and maximum. The default is `range(-4 4)`.

Line

line_options affect the rendition of the plotted IIFs; see [G-3] ***line_options***.

Add plots

`addplot(`*plot*`)` allows adding more `graph twoway` plots to the graph; see [G-3] ***addplot_option***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***).

Data

`n(#)` specifies the number of points at which the IIFs are to be evaluated. The default is `n(300)`.

`data(`*filename*`[ , replace ])` saves the plot data to a Stata data file.

Remarks and examples

`irtgraph iif` plots IIFs after estimating the parameters of an IRT model using `irt`.

In IRT, the term "information" is used to describe reliability or precision of an item or a whole instrument. More reliable items measure the latent trait around the estimated difficulty parameter with greater precision.

IIFs are useful in test development and item evaluation. Depending on the specific needs of the test, items can be chosen to cover the whole spectrum or to focus on a particular range of the ability scale.

The example below shows how to use `irtgraph iif` after a simple 2PL model; see example 4 of [IRT] **irtgraph icc** for remarks on how `irtgraph iif` behaves after a group IRT model.

▷ Example 1: IIF for binary items

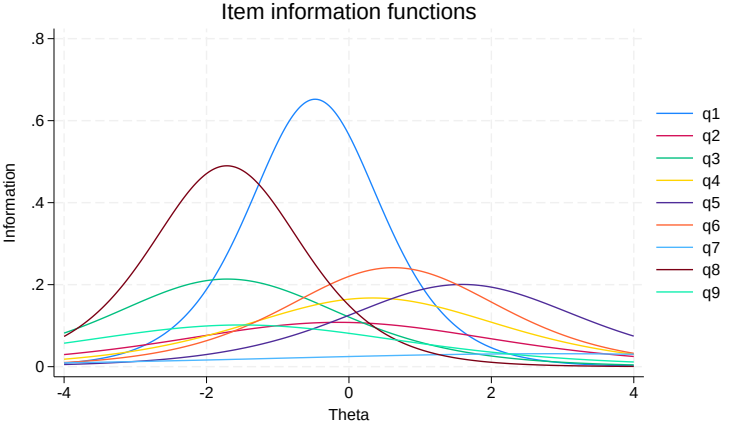
We continue with the 2PL model from example 1 of [IRT] **irt 2pl**. Recall that we fit a 2PL model to the nine binary items.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt 2pl q1-q9
```

(*output omitted*)

Now we plot the IIF for each item in the fitted model.

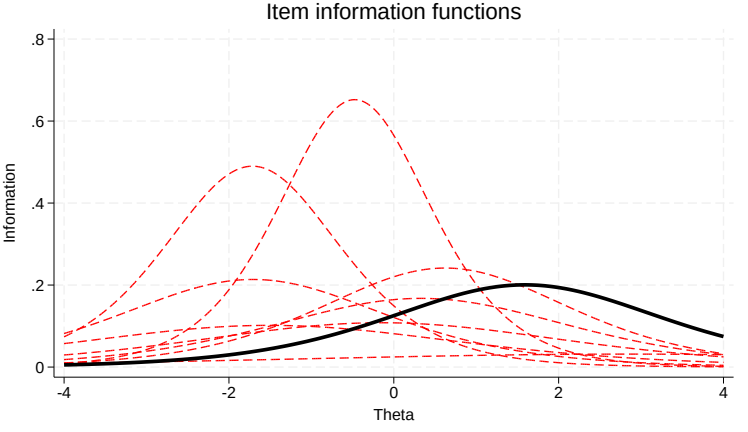
```
. irtgraph iif
```



For binary items, the amount of information is proportional to the discrimination parameter. Items `q1` and `q8` have the two highest discrimination estimates and provide more information than the remaining items. For a 2PL model, the maximum information is provided at $\theta = b_i$.

irtgraph iif's full syntax allows us to apply line styles to each item as we see fit. Let's say we expect more discrimination and therefore more information from a relatively difficult item q5 and thus want q5 to stand out in the IIF plot. To accomplish this, we specify thick and black line styles for q5 to distinguish it from the other items, which we specify with red and dashed line styles.

```
. irtgraph iif (q1-q4 q6-q9, lcolor(red) lpattern(dash))
> (q5, lcolor(black) lwidth(thick)), legend(off)
```



Looking at either IIF graph, we seem to have more item information in the negative region of the latent trait than in the positive region. This suggests that the whole test provides more information about students located at the lower end of the latent trait spectrum, which we show graphically in example 1 of [IRT] **irtgraph tif**.

◁

Methods and formulas

For a given item i with categories $k = 1, \ldots, K$, let $p_{ik}(\theta)$ be the probability of a respondent with latent trait value θ selecting response category k. The functional form of $p_{ik}(\theta)$ depends on the IRT model used to fit item i to the data. The category information function, for category k of item i, is defined as

$$I_{ik}(\theta) = -\frac{\partial^2 \log p_{ik}(\theta)}{\partial \theta^2}$$

The IIF for item i is the sum of its category information functions, weighted by the category probabilities.

$$I_i(\theta) = \sum_{k=1}^{K} I_{ik}(\theta) p_{ik}(\theta)$$

See Birnbaum (1968) and Samejima (1969, 1972, 1977) for a more detailed discussion of item information functions.

References

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

Raciborski, R. 2015. Spotlight on irt. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2015/07/31/spotlight-on-irt/.

Samejima, F. 1969. Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monograph Supplement*, no. 17.

———. 1972. A general model for free-response data. *Psychometrika Monograph Supplement*, no. 18.

———. 1977. Weekly parallel tests in latent trait theory with some criticisms of classical test theory. *Psychometrika* 42: 193–198. https://doi.org/10.1007/BF02294048.

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

[IRT] **irtgraph tif** — Test information function plot

irtgraph tif — Test information function plot

Description

`irtgraph tif` plots the test information function (TIF) for the currently fitted IRT model.

Quick start

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

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

Plot the TIF for the fitted model

```
irtgraph tif
```

Plot the TIF and its standard error

```
irtgraph tif, se
```

Fit a group 2PL model

```
irt 2pl b1-b9, group(female)
```

Plot the TIFs for the fitted model for both groups

```
irtgraph tif
```

Menu

Statistics > IRT (item response theory)

Syntax

```
irtgraph tif [ , options ]
```

options	Description
Plots	
`se[(`*line_options*`)]`	plot the standard error of the TIF
`range(# #)`	plot over $\theta = \#$ to $\#$
Line	
line_options	affect rendition of the plotted TIF
Add plots	
`addplot(`*plot*`)`	add other plots to the TIF plot
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than `by()` documented in [G-3] ***twoway_options***
Data	
`n(#)`	evaluate TIF at # points; default is `n(300)`
`data(`*filename*`[ , replace ])`	save plot data to a file

Options

Plots

`se[(`*line_options*`)]` requests the standard error of the TIF be plotted. The optional *line_options* specify how the lines are rendered; see [G-3] ***line_options***.

`range(# #)` specifies the range of values for θ. This option requires a pair of numbers identifying the minimum and maximum. The default is `range(-4 4)`.

Line

line_options affect the rendition of the plotted TIF; see [G-3] ***line_options***.

Add plots

`addplot(`*plot*`)` allows adding more `graph twoway` plots to the graph; see [G-3] ***addplot_option***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***).

Data

`n(#)` specifies the number of points at which the plotted lines are to be evaluated. The default is `n(300)`.

`data(`*filename*`[ , replace ])` saves the plot data to a Stata data file.

Remarks and examples

`irtgraph tif` plots the TIF after estimating the parameters of an IRT model using `irt`.

In IRT, the term "information" is used to describe reliability or precision of an item or a whole instrument. More reliable instruments measure the latent trait around the estimated difficulty parameter with greater precision.

The TIF is useful in test development where, depending on the specific needs, the test can be chosen to cover the whole spectrum or to focus on a particular range of the ability scale. For tests with alternate formats, TIFs are used to ensure the formats carry the same information across the targeted latent trait range.

Example 1: TIF for binary items

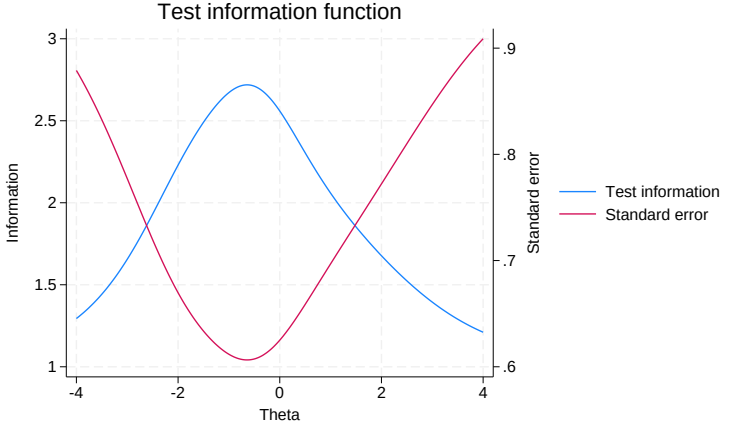
We continue with the 2PL model from example 1 of [IRT] **irt 2pl**. Recall that we fit a 2PL model to the nine binary items.

```
. use https://www.stata-press.com/data/r19/masc1
(Data from De Boeck & Wilson (2004))
. irt 2pl q1-q9
```

(*output omitted*)

In example 1 of [IRT] **irtgraph iif**, we plotted IIFs and noted that items provided more information over the negative range of the latent trait than over the positive range. This asymmetry is also present in the TIF, which we plot below.

```
. irtgraph tif, se
```



The test provides the most information around the latent trait between approximately -1.5 to 0. If we wished for the test to provide more information around a particular latent trait range, we could include additional items that contribute more information within the desired range. Looking at the standard error curve, we observe that the amount of information provided by the test at θ is inversely related to the precision with which ability is estimated at that point.

Methods and formulas

Test information is the negative of the expectation of the second derivative with respect to θ of the log likelihood defined in *Methods and formulas* of [IRT] **irt hybrid**.

$$I(\theta) = -E\left\{\frac{\partial^2}{\partial\theta^2}\log L(\boldsymbol{B})\right\}$$

Given an instrument consisting of I items, the formula above reduces to

$$I(\theta) = \frac{1}{\sigma_\theta^2} + \sum_{i=1}^{I} I_i(\theta)$$

where $I_i(\theta)$ is as defined in *Methods and formulas* of [IRT] **irtgraph iif**. Thus, the TIF is the sum of the individual IIFs and the prior variance of the latent trait, σ_θ^2. `irt` fits IRT models with σ_θ^2 constrained to 1.

The standard error of the TIF is given by

$$\text{se}(\hat{\theta}) = \frac{1}{\sqrt{I(\theta)}}$$

Reference

Raciborski, R. 2015. Spotlight on irt. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2015/07/31/spotlight-on-irt/.

Also see

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

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

[IRT] **irt 2pl** — Two-parameter logistic model

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

[IRT] **irt grm** — Graded response model

[IRT] **irt hybrid** — Hybrid IRT models

[IRT] **irt nrm** — Nominal response model

[IRT] **irt pcm** — Partial credit model

[IRT] **irt rsm** — Rating scale model

[IRT] **irtgraph iif** — Item information function plot

Description Remarks and examples References Also see

Description

Differential item functioning (DIF) occurs when items that are intended to measure a latent trait are unfair, favoring one group of individuals over another. This entry provides an overview of DIF. See the following manual entries for details about the individual DIF tests, including syntax and worked examples.

`diflogistic`	Logistic regression DIF test
`difmh`	Mantel–Haenszel DIF test

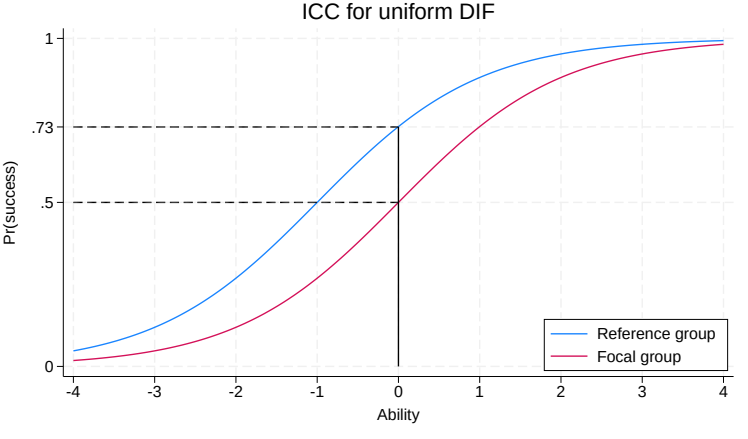
Remarks and examples

DIF is often investigated in conjunction with fitting item response theory (IRT) models. For an introduction to the IRT features in Stata, we encourage you to read [IRT] **irt** first.

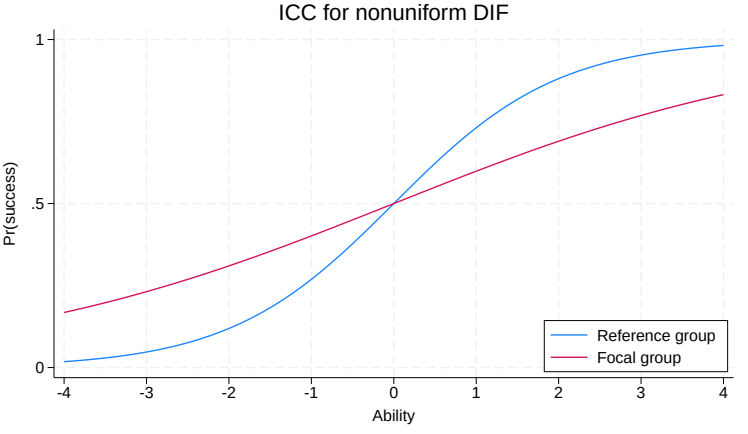
Investigating DIF involves evaluating whether a test item behaves differently across respondents with the same value of the latent trait. An item "functions differently" across individuals with the same latent trait level if these individuals have different probabilities of selecting a given response. A good overview of DIF procedures can be found in Penfield and Camilli (2007) and Osterlind and Everson (2009). Holland and Wainer (1993) provide a thorough treatment of statistical methodologies and practical issues surrounding DIF analysis.

It is convenient to illustrate DIF using item characteristic curves even though many DIF diagnostics are not based on fitting an IRT model. These include the Mantel–Haenszel test available in `difmh` and the logistic regression test available in `diflogistic`.

By convention, the hypothesized disadvantaged individuals are categorized as the focal group, and the advantaged ones are categorized as the reference group. The graph below shows an example of uniform DIF. The item favors the reference group over the entire range of the latent trait (ability). For example, given ability $= 0$, the probability of getting the item right is 0.73 for an individual from the reference group but only 0.50 for an individual from the focal group.



When the two item characteristic curves cross, we have a case of nonuniform DIF; that is to say, different groups are favored in different ranges of the latent trait. In the graph below, the item favors the focal group over the negative range of ability, and the item favors the reference group over the positive range of ability.



The MH test is formally a test of uniform DIF but can, in some cases, detect nonuniform DIF; see [IRT] **difmh** for details.

The logistic regression procedure can detect both uniform and nonuniform DIF; see [IRT] **diflogistic** for details.

References

Holland, P. W., and H. Wainer, eds. 1993. *Differential Item Functioning*. Hillsdale, NJ: Lawrence Erlbaum. https://doi.org/10.4324/9780203357811.

Osterlind, S. J., and H. T. Everson. 2009. *Differential Item Functioning*. 2nd ed. Thousand Oaks, CA: Sage. https://doi.org/10.4135/9781412993913.

Penfield, R. D., and G. Camilli. 2007. "Differential item functioning and item bias". In *Handbook of Statistics*, edited by C. R. Rao and S. Sinharay, vol. 26: 125–167. Amsterdam: Elseiver. https://doi.org/10.1016/S0169-7161(06)26005-X.

Also see

[IRT] **diflogistic** — Logistic regression DIF

[IRT] **difmh** — Mantel–Haenszel DIF

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

diflogistic — Logistic regression DIF

Description

`diflogistic` uses logistic regression to test whether an item exhibits differential item functioning (DIF) between two observed groups. Logistic regression is used to test for both uniform and nonuniform DIF, that is, whether an item favors one group over the other for all values of the latent trait or for only some values of the latent trait.

Quick start

Logistic regression test for binary items `b1` to `b100` using variable `grpvar` to identify the focal and reference groups

```
diflogistic b1-b100, group(grpvar)
```

Same as above, but request the test for items `b5`, `b10`, and `b15` only

```
diflogistic b1-b100, group(grpvar) items(b5 b10 b15)
```

Replay the results, but show only items with p-values ≤ 0.05

```
diflogistic, maxp(.05)
```

Menu

Statistics > IRT (item response theory)

Syntax

```
diflogistic varlist [if] [in] [weight], group(varname) [options]
```

options	Description
Main	
* `group(`*varname*`)`	specify variable that identifies groups
`total(`*varname*`)`	specify total score variable
`items(`*varlist*$_i$`)`	calculate logistic regression test for items in *varlist*$_i$ only
`nolistwise`	do not use listwise deletion to handle missing values
Reporting	
`maxp(#)`	display only items with p-value $\leq$ #
`sformat(%`*fmt*`)`	display format for χ^2 values; default is `sformat(%9.2f)`
`pformat(%`*fmt*`)`	display format for p-values; default is `pformat(%9.4f)`

* `group()` is required.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`fweights` are allowed; see **[U] 11.1.6 weight**.

Options

Main

`group(`*varname*`)` specifies the numeric variable that identifies the focal group and the reference group. The groups should be coded 1 and 0, respectively. `group()` is required.

`total(`*varname*`)` specifies the variable to be used as a total score. By default, the total score is calculated as the row sum of the item variables.

`items(`*varlist*$_i$`)` requests that a logistic regression test be calculated only for the specified items. *varlist*$_i$ must be a subset of *varlist*. By default, the statistics are calculated for all the items in *varlist*.

`nolistwise` specifies to omit observations where all the variables are missing. By default, observations with any missing values are omitted.

Reporting

`maxp(#)` requests that only items with p-value $\leq$ # be displayed.

`sformat(%`*fmt*`)` specifies the display format used for the χ^2 values of the output table. The default is `sformat(%9.2f)`.

`pformat(%`*fmt*`)` specifies the display format used for the p-values of the output table. The default is `pformat(%9.4f)`.

Remarks and examples

The following discussion is about how to use diflogistic to test for uniform and nonuniform DIF in binary items. If you are new to DIF, we encourage you to read [IRT] **DIF** first. If you are new to item response theory or to the item response theory features in Stata, we encourage you to read [IRT] **irt** first.

▷ Example 1: Logistic regression test for uniform and nonuniform DIF

To illustrate logistic regression DIF analysis, we use an abridged version of the mathematics and science data from De Boeck and Wilson (2004). Student responses to nine test items q1–q9 are coded 1 for correct and 0 for incorrect. We want to test for DIF based on sex. Here we tabulate the variable female.

```
. use https://www.stata-press.com/data/r19/masc2
(Data from De Boeck & Wilson (2004))
. tabulate female
```

Female	Freq.	Percent	Cum.
Male	761	50.73	50.73
Female	739	49.27	100.00
Total	1,500	100.00	

We have 761 male and 739 female students in our dataset. The females are coded 1 and represent the focal group.

We perform the logistic regression test on items q1–q9 by typing

```
. diflogistic q1-q9, group(female)
Logistic regression DIF analysis
```

	Nonuniform		Uniform	
Item	chi2	prob.	chi2	prob.
q1	1.03	0.3092	13.20	0.0003
q2	1.39	0.2388	1.80	0.1793
q3	0.39	0.5316	6.90	0.0086
q4	7.25	0.0071	4.89	0.0270
q5	2.29	0.1300	5.91	0.0150
q6	1.18	0.2780	0.43	0.5117
q7	0.04	0.8352	2.61	0.1064
q8	0.96	0.3270	2.24	0.1347
q9	0.23	0.6285	2.23	0.1352

Columns 2 and 3 report the results of logistic regression tests for the presence of nonuniform DIF. Using a 5% significance level, we conclude that only item q4 exhibits nonuniform DIF.

Columns 4 and 5 report the results of logistic regression tests for the presence of uniform DIF. These tests indicate that items q1, q3, and q5 exhibit uniform DIF, again using a 5% significance level. We use this test only for items that do not exhibit nonuniform DIF; thus, in our example, we ignore the uniform DIF test reported for item q4.

A visual examination of the output table becomes cumbersome even for a moderate number of items. We can use `diflogistic` to display only the items whose p-value falls below a certain significance level. Below we redisplay the results with the `maxp(.05)` option.

```
. diflogistic, maxp(.05)
Logistic regression DIF analysis
```

	Nonuniform		Uniform	
Item	chi2	prob.	chi2	prob.
q1	.	.	13.20	0.0003
q3	.	.	6.90	0.0086
q4	7.25	0.0071	.	.
q5	.	.	5.91	0.0150

Now it is much easier to see that item `q4` exhibits a nonuniform DIF and items `q1`, `q3`, and `q5` exhibit uniform DIF.

Note that neither test reports anything about the amount or direction of DIF exhibited by an item. For items that exhibit uniform DIF, you can use a common odds ratio reported by `difmh` to assess the amount and direction of DIF.

Stored results

`diflogistic` stores the following in `r()`:

Macros

`r(cmd)`	`diflogistic`
`r(cmdline)`	command as typed
`r(items)`	names of items
`r(wtype)`	weight type
`r(wexp)`	weight expression
`r(group)`	group variable
`r(total)`	name of alternative total score variable, if specified

Matrices

`r(dif)`	results table
`r(_N)`	number of observations per item

Methods and formulas

Let L_1, L_2, and L_3 be the log-likelihood values associated with the following models, respectively,

$$\text{logit}\{\Pr(\mathbf{y})\} = \tau_0 + \tau_1\mathbf{t} + \tau_2\mathbf{g} + \tau_3(\mathbf{t}\times\mathbf{g}) \tag{1}$$

$$\text{logit}\{\Pr(\mathbf{y})\} = \tau_0 + \tau_1\mathbf{t} + \tau_2\mathbf{g} \tag{2}$$

$$\text{logit}\{\Pr(\mathbf{y})\} = \tau_0 + \tau_1\mathbf{t} \tag{3}$$

where $\mathbf{y}$ is a vector of responses for a given item; $\mathbf{t}$ is the latent trait, most commonly represented by the observed total score; and $\mathbf{g}$ is a dichotomous variable representing the focal group.

Likelihood-ratio tests are used to compare the nested models. The test for nonuniform DIF compares models (1) and (2) and is given by $\text{LR}_1 = -2(L_1 - L_2)$. LR_1 is distributed as χ^2_1. If the null hypothesis of no nonuniform DIF is rejected, we do not proceed to the test for uniform DIF.

The test for uniform DIF compares models (2) and (3) and is given by $\text{LR}_2 = -2(L_2 - L_3)$. LR_2 is distributed as χ^2_1.

References

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Swaminathan, H., and H. J. Rogers. 1990. Detecting differential item functioning using logistic regression procedures. *Journal of Educational Measurement* 27: 361–370. https://doi.org/10.1111/j.1745-3984.1990.tb00754.x.

Also see

[IRT] **DIF** — Introduction to differential item functioning

[IRT] **difmh** — Mantel–Haenszel DIF

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

difmh — Mantel–Haenszel DIF

Description

`difmh` calculates the Mantel–Haenszel (MH) χ^2 and common odds ratio for dichotomously scored items. The MH statistics are used to determine whether an item exhibits uniform differential item functioning (DIF) between two observed groups, that is, whether an item favors one group relative to the other for all values of the latent trait.

Quick start

MH χ^2 and common odds ratio for binary items `b1` to `b100` using variable `grpvar` to identify the focal and reference groups

```
difmh b1-b100, group(grpvar)
```

Same as above, but request MH statistics for items `b5`, `b10`, and `b15` only

```
difmh b1-b100, group(grpvar) items(b5 b10 b15)
```

Replay the results, but show only items with p-values ≤ 0.05

```
difmh, maxp(.05)
```

Menu

Statistics > IRT (item response theory)

Syntax

```
difmh varlist [if] [in] [weight], group(varname) [options]
```

options	Description
Main	
* `group(`*varname*`)`	specify variable that identifies groups
`total(`*varname*`)`	specify total score variable
`items(`*varlist*$_i$`)`	calculate MH statistics for items in *varlist*$_i$ only
`nolistwise`	do not use listwise deletion to handle missing values
`noyates`	do not apply Yates's correction for continuity; default is to apply the continuity correction
Reporting	
`level(`#`)`	set confidence level; default is `level(95)`
`maxp(`#`)`	display only items with p-value $\leq$ #
`sformat(`*%fmt*`)`	display format for χ^2 values; default is `sformat(%9.2f)`
`pformat(`*%fmt*`)`	display format for p-values; default is `pformat(%9.4f)`
`oformat(`*%fmt*`)`	display format for odds-ratio statistics; default is `oformat(%9.4f)`

* `group()` is required.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`fweights` are allowed; see **[U] 11.1.6 weight**.

Options

Main

`group(`*varname*`)` specifies the numeric variable that identifies the focal group and the reference group. The groups should be coded 1 and 0, respectively. `group()` is required.

`total(`*varname*`)` specifies the variable to be used as a total score. By default, the total score is calculated as the row sum of the item variables.

`items(`*varlist*$_i$`)` requests that MH statistics be calculated only for the specified items. *varlist*$_i$ must be a subset of *varlist*. By default, the statistics are calculated for all the items in *varlist*.

`nolistwise` specifies to omit observations where all the variables are missing. By default, observations with any missing values are omitted.

`noyates` specifies that Yates's correction for continuity not be applied when calculating the MH χ^2 statistic. By default, the continuity correction is applied.

Reporting

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

`maxp(#)` requests that only items with p-value $\leq$ # be displayed.

`sformat(`*%fmt*`)` specifies the display format used for the χ^2 values of the output table. The default is `sformat(%9.2f)`.

`pformat(`*%fmt*`)` specifies the display format used for the p-values of the output table. The default is `pformat(%9.4f)`.

`oformat(`*%fmt*`)` specifies the display format used for the odds-ratio statistics of the output table. The default is `oformat(%9.4f)`.

Remarks and examples

The following discussion is about how to use `difmh` to test for uniform DIF in binary items. If you are new to DIF, we encourage you to read [IRT] **DIF** first. If you are new to item response theory or to the item response theory features in Stata, we encourage you to read [IRT] **irt** first.

▷ Example 1: MH test of uniform DIF

To illustrate the MH DIF analysis, we use an abridged version of the mathematics and science data from De Boeck and Wilson (2004). Student responses to nine test items `q1`–`q9` are coded 1 for correct and 0 for incorrect. We want to test for DIF based on sex. Here we tabulate the variable `female`.

```
. use https://www.stata-press.com/data/r19/masc2
(Data from De Boeck & Wilson (2004))
. tabulate female
```

Female	Freq.	Percent	Cum.
Male	761	50.73	50.73
Female	739	49.27	100.00
Total	1,500	100.00	

We have 761 male and 739 female students in our dataset. The females are coded 1 and represent the focal group.

We perform the MH procedure on items `q1`–`q9` by typing

```
. difmh q1-q9, group(female)
Mantel-Haenszel DIF analysis
```

Item	chi2	Prob.	Odds ratio	[95% conf.	interval]
q1	12.47	0.0004	1.6053	1.2395	2.0790
q2	1.79	0.1813	1.1809	0.9354	1.4907
q3	6.58	0.0103	1.4543	1.0993	1.9238
q4	3.86	0.0496	0.7879	0.6241	0.9947
q5	5.00	0.0253	0.7011	0.5189	0.9472
q6	0.49	0.4835	1.1046	0.8567	1.4243
q7	1.77	0.1836	0.8359	0.6500	1.0750
q8	2.09	0.1478	0.7761	0.5615	1.0727
q9	2.03	0.1546	0.8294	0.6479	1.0618

The `chi2` and `prob.` columns contain the MH χ^2 statistic with the associated significance level. Items `q1`, `q3`, `q4`, and `q5` exhibit DIF based on a 5% significance level. However, significant statistics do not tell us anything about the amount or direction of DIF exhibited by an item.

The last three columns present the MH common odds ratio with the associated confidence interval. A common odds ratio greater than 1 indicates DIF in favor of the focal group. The results suggest that items `q1` and `q3` favor females, and items `q4` and `q5` favor males.

A visual examination of the output table becomes cumbersome even for a moderate number of items. We can ask `difmh` to display only items whose p-value falls below a certain significance level. Below we redisplay the results with the `maxp(.05)` option.

```
. difmh, maxp(.05)
Mantel-Haenszel DIF analysis
```

Item	chi2	Prob.	Odds ratio	[95% conf.	interval]
q1	12.47	0.0004	1.6053	1.2395	2.0790
q3	6.58	0.0103	1.4543	1.0993	1.9238
q4	3.86	0.0496	0.7879	0.6241	0.9947
q5	5.00	0.0253	0.7011	0.5189	0.9472

Stored results

`difmh` stores the following in `r()`:

Scalars

`r(N)`	number of observations
`r(level)`	significance level
`r(yates)`	1 if Yates's continuity correction is used, 0 otherwise

Macros

`r(cmd)`	`difmh`
`r(cmdline)`	command as typed
`r(items)`	names of items
`r(wtype)`	weight type
`r(wexp)`	weight expression
`r(group)`	group variable
`r(total)`	name of alternative total score variable, if specified

Matrices

`r(dif)`	results table
`r(sigma2)`	estimated variance of the common odds ratio
`r(_N)`	number of observations per item

Methods and formulas

The MH test, also known as the Cochran–Mantel–Haenszel test, is used to determine whether two dichotomous variables are independent of one another after conditioning on a third variable; see Mantel and Haenszel (1959) and Holland and Thayer (1988) for details.

In item response theory, one dichotomous variable represents the reference and the focal group, and the other represents a response to an item scored as correct and incorrect. The conditioning variable is the latent trait, most commonly represented by the observed total score. For a dichotomously scored instrument of length K, the total score ranges from 0 to K.

The MH χ^2 statistic is based on the sum of the 2×2 contingency tables calculated for each value (stratum) of the total score. A single 2×2 table for the kth score is

	Correct	Incorrect	Total
Reference	n_{11k}	n_{12k}	$n_{1.k}$
Focal	n_{21k}	n_{22k}	$n_{2.k}$
Total	$n_{.1k}$	$n_{.2k}$	n_k

Incomplete contingency tables do not contribute to the MH statistic; this includes perfect scores and zero scores. For items with fewer than two complete contingency tables, `difmh` reports missing values.

The MH χ^2 statistic is given as

$$\text{MH}_{\chi^2} = \frac{\left[\left|\sum_{k=1}^{K-1}\{n_{11k} - E(n_{11k})\}\right| - c\right]^2}{\sum_{k=1}^{K-1} \text{var}(n_{11k})}$$

where

$$E(n_{11k}) = \frac{n_{1.k}n_{.1k}}{n_k}$$

$$\text{var}(n_{11k}) = \frac{n_{1.k}n_{2.k}n_{.1k}n_{.2k}}{n_k^2(n_k - 1)}$$

and $c = 0$ when option `noyates` is specified; $c = 1/2$ otherwise.

The statistic is evaluated against a χ^2 distribution with one degree of freedom. A significant MH χ^2 statistic suggests the presence of DIF in an item, however, the statistic does not indicate the amount of DIF.

To assess the amount of DIF in an item, we can use the MH common odds-ratio (OR) statistic. The statistic is given as

$$\text{OR}_{\text{MH}} = \frac{\sum_{k=1}^{K-1} n_{11k}n_{22k}/n_k}{\sum_{k=1}^{K-1} n_{12k}n_{21k}/n_k}$$

Under the null hypothesis of no DIF, $\text{OR}_{\text{MH}} = 1$.

The confidence interval for OR_{MH} is based on the variance estimate of $\log(\text{OR}_{\text{MH}})$ proposed by Robins, Breslow, and Greenland (1986) and Phillips and Holland (1987):

$$\begin{aligned}
\hat{\sigma}^2 &= \widehat{\text{var}}\{\log(\text{OR}_{\text{MH}})\} \\
&= \frac{\sum_{k=1}^{K-1}(n_{11k} + n_{22k})(n_{11k}n_{22k})/n_k^2}{2(\sum_{k=1}^{K-1} n_{11k}n_{22k}/n_k)^2} \\
&+ \frac{\sum_{k=1}^{K-1}\{(n_{11k} + n_{22k})(n_{12k}n_{21k}) + (n_{12k} + n_{21k})(n_{11k}n_{22k})\}/n_k^2}{2(\sum_{k=1}^{K-1} n_{11k}n_{22k}/n_k)(\sum_{k=1}^{K-1} n_{12k}n_{21k}/n_k)} \\
&+ \frac{\sum_{k=1}^{K-1}(n_{12k} + n_{21k})(n_{12k}n_{21k})/n_k^2}{2(\sum_{k=1}^{K-1} n_{12k}n_{21k}/n_k)^2}
\end{aligned}$$

The $100(1 - \alpha/2)\%$ confidence interval for OR_{MH} is then given by

$$[\text{OR}_{\text{MH}} \times \exp(-z\hat{\sigma}),\ \text{OR}_{\text{MH}} \times \exp(z\hat{\sigma})]$$

References

De Boeck, P., and M. Wilson, eds. 2004. *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer. https://doi.org/10.1007/978-1-4757-3990-9.

Holland, P. W., and D. T. Thayer. 1988. "Differential item performance and the Mantel–Haenszel procedure". In *Test Validity*, edited by H. Wainer and H. I. Braun, 129–145. Hillsdale, NJ: Lawrence Erlbaum.

Mantel, N., and W. Haenszel. 1959. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the National Cancer Institute* 22: 719–748. Reprinted in *Evolution of Epidemiologic Ideas: Annotated Readings on Concepts and Methods*, ed. S. Greenland, pp. 112–141. Newton Lower Falls, MA: Epidemiology Resources.

Phillips, A., and P. W. Holland. 1987. Estimators of the variance of the Mantel–Haenszel log-odds-ratio estimate. *Biometrics* 43: 425–431. https://doi.org/10.2307/2531824.

Robins, J. M., N. E. Breslow, and S. Greenland. 1986. Estimators of the Mantel–Haenszel variance consistent in both sparse data and large-strata limiting models. *Biometrics* 42: 311–323. https://doi.org/10.2307/2531052.

Also see

[IRT] **DIF** — Introduction to differential item functioning

[IRT] **diflogistic** — Logistic regression DIF

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

Glossary

1PL. See *one-parameter logistic model*.

2PL. See *two-parameter logistic model*.

3PL. See *three-parameter logistic model*.

ability. See *latent trait*.

BCC. See *boundary characteristic curve*.

binary item. A binary item is an item that is scored as either 0 or 1.

boundary characteristic curve. A boundary characteristic curve (BCC) expresses the probability of transitioning across a given boundary threshold that separates the ordered item categories into two groups as a function of the latent trait.

calibration. The procedure of estimating parameters of an IRT model.

categorical item. A categorical item is an item that is either ordinal or nominal.

category boundary curve. See *boundary characteristic curve*.

category boundary location. See *difficulty*.

category characteristic curve. A category characteristic curve (CCC) expresses the probability of a response in a given item category as a function of the latent trait.

category response function. See *category characteristic curve*.

CCC. See *category characteristic curve*.

conditional independence. The assumption that responses are not correlated after controlling for the latent trait.

dichotomous item. See *binary item*.

DIF. See *differential item functioning*.

differential item functioning. Differential item functioning involves evaluating whether a test item behaves differently between groups, after the groups have been matched on the latent trait.

difficulty. A level of the latent trait needed to pass an item or an item category.

discrimination. A measure of how well an item can distinguish between contiguous latent trait levels near the inflection point of an item characteristic curve.

empirical Bayes. In IRT models, empirical Bayes refers to the method of prediction of the latent trait after the model parameters have been estimated. The empirical Bayes method uses Bayesian principles to obtain the posterior distribution of the latent trait. However, instead of assuming a prior distribution for the model parameters, one treats the parameters as given.

Gauss–Hermite quadrature. In the context of IRT models, Gauss–Hermite quadrature (GHQ) is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individuals are fixed during the optimization process.

generalized partial credit model. The generalized partial credit model (GPCM) is an IRT model for ordinal responses. The categories within each item vary in their difficulty and share the same discrimination parameter.

GHQ. See *Gauss–Hermite quadrature*.

GPCM. See *generalized partial credit model*.

graded response model. The graded response model (GRM) is an extension of the two-parameter logistic model to ordinal responses. The categories within each item vary in their difficulty and share the same discrimination parameter.

GRM. See *graded response model*.

guessing. The guessing parameter incorporates the impact of chance on an observed response. The parameter lifts the lower asymptote of the item characteristic curve above zero.

hybrid model. A hybrid IRT model is a model that performs a single calibration of an instrument consisting of different response formats.

ICC. See *item characteristic curve*.

IIF. See *item information function*.

information. Precision with which an item or an instrument measures the latent trait; also see *item information function* and *test information function*.

instrument. A collection of items, usually called a test, a survey, or a questionnaire.

invariance. When an IRT model fits the data exactly in the population, then the estimated item parameters should be the same, within sampling error, regardless of what sample the data were derived from, and the estimated person latent traits should be the same regardless of what items they are based on.

IRT. See *item response theory*.

item. An item is a single question or task on a test or an instrument.

item characteristic curve. An item characteristic curve (ICC) expresses the probability for a given response to a binary item as a function of the latent trait.

item information function. An item information function (IIF) indicates the precision of an item along the latent trait continuum.

item location. Location of an item on the difficulty scale.

item response function. See *item characteristic curve*.

item response theory. Item response theory (IRT) is a theoretical framework organized around the concept of the latent trait. IRT encompasses a set of models and associated statistical procedures that relate observed responses on an instrument to a person's level of the latent trait.

latent space. Number of latent traits that are measured by an instrument. All IRT models described in this manual assume a unidimensional latent space or, in other words, that a single latent trait explains the response pattern.

latent trait. A variable or construct that cannot be directly observed.

local independence. See *conditional independence*.

lower asymptote. See *guessing*.

MCAGHQ. See *mode-curvature adaptive Gauss–Hermite quadrature*.

mean–variance adaptive Gauss–Hermite quadrature. In the context of IRT models, mean–variance adaptive Gauss–Hermite quadrature (MVAGHQ) is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individuals are updated during the optimization process by using the posterior mean and the posterior standard deviation.

mode-curvature adaptive Gauss–Hermite quadrature. In the context of IRT models, mode-curvature adaptive Gauss–Hermite quadrature (MCAGHQ) is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individuals are updated during the optimization process by using the posterior mode and the standard deviation of the normal density that approximates the log posterior at the mode.

MVAGHQ. See *mean–variance adaptive Gauss–Hermite quadrature*.

nominal item. A nominal item is an item scored in categories that have no natural ordering.

nominal response model. The nominal response model (NRM) is an IRT model for nominal responses. The categories within each item vary in their difficulty and discrimination.

NRM. See *nominal response model*.

one-parameter logistic model. The one-parameter logistic (1PL) model is an IRT model for binary responses where items vary in their difficulty but share the same discrimination parameter.

operating characteristic curve. See *category characteristic curve*.

ordinal item. An ordinal item is an item scored on a scale where a higher score indicates a "higher" outcome.

partial credit model. The partial credit model (PCM) is an IRT model for ordinal responses. The categories across all items vary in their difficulty and share the same discrimination parameter.

PCM. See *partial credit model*.

person location. Location of a person on the latent trait scale.

polytomous item. See *categorical item*.

posterior mean. In IRT models, posterior mean refers to the predictions of the latent trait based on the mean of the posterior distribution.

posterior mode. In IRT models, posterior mode refers to the predictions of the latent trait based on the mode of the posterior distribution.

quadrature. Quadrature is a set of numerical methods to evaluate a definite integral.

rating scale model. The rating scale model (RSM) is an IRT model for ordinal responses. The categories within each item vary in their difficulty; however, the distances between adjacent difficulty parameters are constrained to be the same across the items. The categories across all items share the same discrimination parameter.

RSM. See *rating scale model*.

slope. See *discrimination*.

TCC. See *test characteristic curve*.

test characteristic curve. A test characteristic curve (TCC) is the sum of item characteristic curves and represents the expected score on the instrument.

test information function A test information function (TIF) is the sum of item information functions and indicates the precision of the entire instrument along the latent trait continuum.

three-parameter logistic model. The three-parameter logistic (3PL) model is an IRT model for binary responses where items vary in their difficulty and discrimination and can share or have their own guessing parameter.

TIF. See *test information function*.

total characteristic curve. See *test characteristic curve*.

total information function. See *test information function*.

two-parameter logistic model. The two-parameter logistic (2PL) model is an IRT model for binary responses where items vary in their difficulty and discrimination.

unidimensionality. See *latent space*.

Subject and author index

See the combined subject index and the combined author index in the *Stata Index*.