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**.

`fweight`s, `iweight`s, and `pweight`s 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 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, $\mathbf{d} = (d_1, \dots, d_K)$, 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.

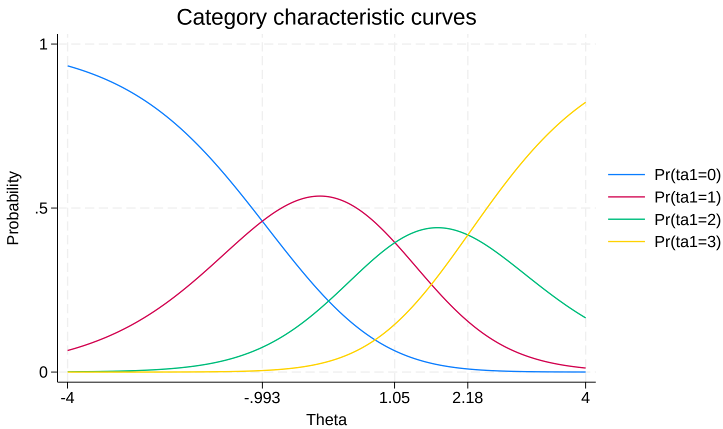
```
. use https://www.stata-press.com/data/r19/charity
(Data from Zheng & Rabe-Hesketh (2007))

. 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.

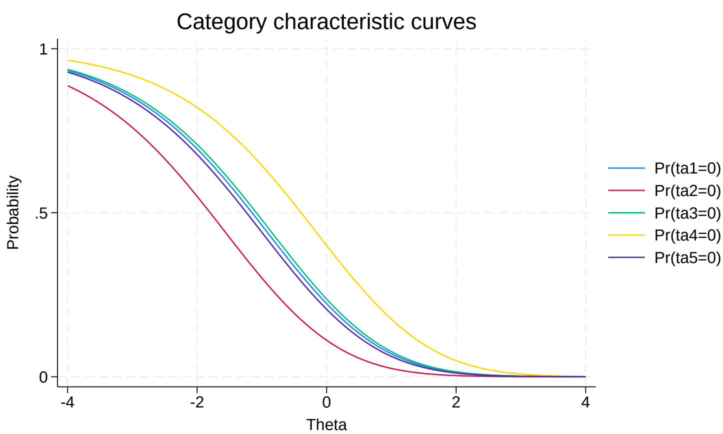
We can show this graphically using CCCs. The curves trace the probability of choosing each category as a function of θ using the estimated RSM parameters. Here we plot the probabilities for item `ta1` using `irtgraph icc`; see [IRT] **irtgraph icc** for details.

```
. 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**