

Package ‘LSMjml’

September 28, 2026

Type Package

Title Fitting Latent Space Item Response Models using Joint Maximum Likelihood Estimation

Version 0.7.0

Author Dylan Molenaar [aut, cre]

Maintainer Dylan Molenaar <d.molenaar@uva.nl>

Description In Latent Space Item Response Models, subjects and items are embedded in a multidimensional Euclidean latent space.

As such, interactions among persons, items, and person-item combinations can be revealed that are unmodelled in

more conventional item response theory models. This package implements the meth-

ods from Molenaar & Jeon (2026)<[doi:10.1017/psy.2025.10068](https://doi.org/10.1017/psy.2025.10068)> and can be used to fit La-

tent Space Item Response Models

to data using joint maximum likelihood estimation. The package can handle binary data, ordinal data, and data with mixed scales.

The package incorporates facilities for data simulation, rotation of the latent space, and K-

fold cross-validation to select the

number of dimensions of the latent space.

License GPL-3

Encoding UTF-8

Imports Rcpp (>= 1.0.12), lavaan, pROC, psych

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-09-28 15:10:02 UTC

Contents

LSMboot	2
LSMdists	4
LSMfit	5
LSMrotate	10

LSMselect	12
LSMsim	14
Index	17

LSMboot	<i>Bootstrap Standard Errors for Item Parameters of a Fitted Latent Space Item Response Model</i>
---------	---

Description

This function computes nonparametric case-resampling bootstrap standard errors for the item parameters of a Latent Space Item Response Model (LSIRM) previously fit with [LSMfit](#).

Usage

```
LSMboot(object, X, nboot=100, tol=NULL, ...)
```

Arguments

object	An object of class <code>LSMfit</code> , as returned by LSMfit .
X	The same N by n data matrix that was used to obtain object with LSMfit .
nboot	The number of bootstrap replications. Default is 100.
tol	Convergence criterion passed to LSMfit for each bootstrap replication (see LSMfit). If NULL (the default), a value of .1 is used.
...	Currently not used.

Details

LSMboot implements a nonparametric case-resampling bootstrap for the LSIRM. In each of nboot replications, the N respondents (rows) of X are resampled with replacement, the model is refit to the resampled data with [LSMfit](#) (using the parameter estimates in object as starting values, via object\$as_starts, and the same penalty and C that were used to obtain object), and the resulting parameter estimates are stored. After all replications, the standard deviation of the bootstrap estimates for each item parameter is taken as its bootstrap standard error.

LSMboot only returns standard errors for the item parameters, se_w and se_b, not for the person parameters. Because respondents, not items, are resampled, item identity is preserved across all replications (item *i* in the resampled data is always the same item *i* as in X), so the bootstrap standard deviation of an item's estimates across replications is a meaningful standard error. This is not the case for the person parameters: since a given row of the resampled data corresponds to a different, essentially arbitrary, original respondent in every replication, and LSMboot does not track which original respondent contributed to which row across replications, the standard deviation of the estimates at a given row position across replications would not estimate the sampling uncertainty of any specific respondent's parameters. For this reason, person-level standard errors are not returned by LSMboot.

If, for some item, a bootstrap replication happens to not include any respondent who used that item's highest response category (which can occur for items with a rarely used top category), LSMboot automatically reduces the number of category parameters estimated for that item in that replication. The corresponding entries of output (and, hence, of se_b) are NA for that replication.

Value

A list with values

se_w	A n by R matrix of bootstrap standard errors for the item coordinates w_{ir} .
se_b	A vector of bootstrap standard errors for the item category parameters b_i .
output	The nboot by $(R \times (N+n) + N + \text{length}(\text{object}\$b))$ matrix of raw bootstrap parameter estimates (item coordinates, person coordinates, item category parameters, and person parameters, in that order), from which se_w and se_b are computed.

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

See Also

[LSMfit](#) for fitting LSIRM models.

Examples

```
set.seed(1111)
N=200
nit=15
ndim_z=2
dat_obj=LSMsim(N,nit,ndim_z)
X=dat_obj$X

#fit model
results=LSMfit(X,2)

#bootstrap the item parameters (use more than 20 replications in practice)
boot_results=LSMboot(results,X,nboot=20)

#bootstrap standard errors for the item coordinates
boot_results$se_w
```

LSMdist

*Compute Latent Space Distances from a Fitted Latent Space Item Response Model***Description**

This function computes Euclidean distances in the fitted latent space of an object of class `LSMfit`: between persons and items (the quantity used directly by the model), and, optionally, between persons, or between items.

Usage

```
LSMdist(object, type = c("person-item", "person-person", "item-item", "all"))
```

Arguments

<code>object</code>	An object of class <code>LSMfit</code> , as returned by LSMfit .
<code>type</code>	Character string indicating which distances to return: "person-item" (the default), "person-person", "item-item", or "all". See Details .

Details

The person-item distances, $d(z_p, w_i)$, are the quantity used directly by the LSIRM (see [LSMfit](#)): $g(\pi_{pic}) = \theta_p + \beta_{ic} - d(z_p, w_i)$. `LSMdist` computes these, by default, as the Euclidean distance between every person's coordinates in `object$z` and every item's coordinates in `object$w`.

Although only person-item distances are used in fitting the model, the common Euclidean metric across persons and items means distances between two persons, or between two items, can also be meaningfully computed from the same coordinates, even though these are not explicitly part of the model (see [LSMfit](#)). `type="person-person"` and `type="item-item"` return these, and `type="all"` returns all three as a list.

For `ndim_z=0`, all distances are 0, since there is no latent space to compute a distance in.

Value

If `type` is "person-item", "person-person", or "item-item", a matrix of Euclidean distances (N by n, N by N, or n by n, respectively, where N is the number of persons and n the number of items). If `type="all"`, a list with values

<code>person_item</code>	The N by n matrix of person-item distances.
<code>person_person</code>	The N by N matrix of person-person distances.
<code>item_item</code>	The n by n matrix of item-item distances.

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

See Also

[LSMfit](#) for fitting LSIRM models.

Examples

```
set.seed(1111)
N=1000
nit=20
ndim_z=2
dat_obj=LSMsim(N,nit,ndim_z)
X=dat_obj$X

#fit model
results=LSMfit(X,2)

#the N by n matrix of person-item distances used directly by the model
D=LSMdlist(results)

#for a given person, the items ordered from closest (most "as expected")
#to farthest (most challenging relative to theta and beta)
order(D[1,])

#distances between items, inferred (not directly modeled) from the same
#coordinates
D_items=LSMdlist(results,type="item-item")
```

LSMfit

Fitting Latent Space Item Response Models using Joint Maximum Likelihood Estimation

Description

This function fits a Latent Space Item Response Model (LSIRM) with an R dimensional latent space using penalized Joint Maximum Likelihood (pJML) or constrained Joint Maximum Likelihood (cJML) to observed binary or ordinal item scores.

Usage

```
LSMfit(X, ndim_z, penalty=NULL, C=NULL, starts="default",
       tol=.1e-2, silent=FALSE)
```

Arguments

X	A matrix of size N by n containing the binary or ordinal item scores, where N is the number of subjects and n is the number of items. The number of item score categories can be different across items as long as the lowest score is coded 0 for all items. NA's are allowed.
---	---

<code>ndim_z</code>	Number of dimensions of the latent space, R .
<code>penalty</code>	The weight for the L2 penalty of pJML. If both <code>penalty</code> and <code>C</code> (see below) are NULL (the default), a pJML is used with a weight of 1 (i.e., standard normal prior on all parameters).
<code>C</code>	The tuning parameter for cJML that determines the maximum size (Euclidean norm) of the person and item parameter vectors (see Details). <code>C</code> is not itself the maximum norm: the actual maximum norms for the person and item parameter vectors are derived from <code>C</code> (and from <code>ndim_z</code> and the number of response categories), as described in Details .
<code>starts</code>	Either a list containing starting values for the model parameters (see details), a character string indicating the method of starting value calculation, or a single positive integer indicating the number of random starting sets to try (see details). The character string options are: "default" starting values are determined by first attempting "ml" (see below); if this does not succeed, "wls" is attempted next; if that also does not succeed, 20 random starting sets are tried, in the same way as when <code>starts</code> is set to a positive integer (see below and details). Messages are printed to the screen indicating which step is used, unless <code>silent=TRUE</code>. This is the default. "wls" the starting values are determined by fitting a $R+1$ dimensional item factor model directly to the ordinal item scores in X, using a weighted least squares estimator for ordered data (<code>lavaan::cfa</code> with <code>ordered=TRUE</code>), which relies on polychoric correlations internally as part of that single estimation step "ml" the starting values are determined by first computing the polychoric correlation matrix of X (via <code>psych::polychoric</code>), and then fitting a $R+1$ dimensional linear factor model to that correlation matrix using normal theory maximum likelihood "mds" The starting values are determined via (metric) multidimensional scaling: an SVD is conducted on the double-centered data matrix. If a single positive integer k is given instead (e.g., <code>starts=10</code>), k independent random starting sets are drawn and each is used to fit the model to a loose tolerance of 1 (i.e., <code>tol=1</code>); the starting set that attains the highest log-likelihood among these k preliminary fits is then used as the starting values for the actual model fit, run at the tolerance given in argument <code>tol</code> (see details).
<code>tol</code>	Convergence criterion: Iterations stop if the difference in loglikelihoods between two subsequent iterations is smaller than this number. Default is .001.
<code>silent</code>	Logical. If FALSE, iterations details are printed to the screen during estimation.

Details

LSMfit optimizes the joint likelihood function of the LSIRM described in Molenaar and Jeon (in press) using a variant of the alternating optimization algorithm by Chen et al. (2019) and using either a L2 regularization penalty similar to Bergner et al. (2022) or a constraint on the norms of the parameter vectors similar to Chen et al. (2019). For binary X_{pi} , the LISRM by Molenaar and Jeon is given by:

$$\text{logit}(E(X_{pi})) = \theta_p + b_i - (\sum_{r=1}^R (z_{pr} - w_{ir})^2)^{1/2}$$

where θ_p is a person intercept, b_i is an item intercept, R denotes the dimension of the latent space, and z_{pr} and w_{ir} are respectively the person and item coordinates in the latent space. The matrix $\mathbf{W}$ containing the w_{ir} parameters is constrained to an echelon structure (i.e., all elements from the upper triangle of submatrix $\mathbf{W}_{1:(R-1),1:(R-1)}$ are fixed to 0. Next, cJML estimation involves constraining the Euclidean norm of the person parameter vector $\tau_{1p} = [\theta_p, z_{p1}, z_{p2}, \dots, z_{pR}]$ to be at most C_1 , and the Euclidean norm of the item parameter vector $\tau_{2i} = [b_{i0}, b_{i1}, \dots, b_{i(C_i-1)}, w_{i1}, w_{i2}, \dots, w_{iR}]$ to be at most C_2 . The bounds C_1 and C_2 are not set directly by the user; instead, they are both derived from the single tuning parameter C as

$$C_1 = \frac{1}{2}C\sqrt{R-1}$$

$$C_2 = C\sqrt{R + C_i - 1}$$

where C_i is the number of response categories of item i . That is, C tunes how large the person and item parameter vectors are allowed to be; larger values of C correspond to weaker (less restrictive) constraints. On the contrary, pJML estimation involves adding an L2 regularization penalty for all parameters to the joint likelihood function in such a way that the penalty parameter can be interpreted as the precision of a normal prior on the parameters.

Using the `starts` argument, starting values can be provided in a list containing entries:

`z0` a n by R matrix with starting values for z_{pr}
`w0` a n by R matrix with starting values for w_{ir}
`b0` a n by 1 matrix with starting values for b_i
`theta0` a n by 1 matrix with starting values for θ_p

Alternatively, starting values can be automatically determined by `LSMfit`. To this end, the following $R+1$ factor model will be fit (omitting the item intercept):

$$g(E(X_{pi})) = \eta_{p0} + \sum_{r=1}^R \lambda_{ir} \eta_{pr}$$

where the n by R matrix of λ_{ir} parameters follows the echelon structure above, and $g(\cdot)$ is either the identity link or the probit link (see below). For "wls", the model above is fit directly to the ordinal item scores in $\mathbf{X}$ using a weighted least squares (WLS) estimator for ordered data with a probit link for $g(\cdot)$, which relies on polychoric correlations internally as part of that single estimation step. For "ml", the polychoric correlation matrix of $\mathbf{X}$ is computed first, and the model above is then fit to this matrix using normal theory maximum likelihood (ML) estimation with an identity link for $g(\cdot)$ (i.e., the polychoric correlation matrix is treated as a product-moment correlation matrix). In both cases, the thresholds of the polychoric correlation matrix are taken as the basis for the starting values of b_i , the factor score estimates of η_{p0} are taken as starts for θ_p , the estimates of η_{pr} are taken as the starts for z_{pr} , and the estimates of λ_{ir} are taken as a basis for the starting values of w_{ir} . The WLS approach is statistically the most rigorous approach but can be time consuming, while the ML approach is an ad-hoc approach but which is fast and turns out to work well in practice. When `starts="ml"` or `starts="wls"` is requested explicitly, and especially for models with $R > 2$, fitting the factor model above may fail. In that case, `LSMfit` automatically substitutes a single set of random starting values for the requested method, with a warning.

The default, `starts="default"`, uses a more thorough search instead of relying on this single automatic substitution. First, "ml" is attempted. If it does not succeed, `LSMfit` switches to "wls". If "wls" does not succeed either, `LSMfit` switches to trying 20 random starting sets, in the same

way as when `starts` is set to a positive integer (see below). Messages describing each switch (e.g. `starts = "ml"` did not succeed; switching to `starts = "wls"`) are printed to the screen unless `silent=TRUE`.

`starts` can also be set to a positive integer k (e.g., `starts=10`), in which case k independent random starting sets are generated (in the same way as in the default above) and each is used to fit the model with a tolerance of 1. The candidate that reaches the highest log-likelihood is then used as the starting values for a final fit at the tolerance specified in `tol`. Ordinal items are internally accommodated by dummy coding the items with more than 2 score levels into $C-1$ binary variables using a binary Guttman expansion (see Tutz, 2020). Next, the dummy coded variables are submitted to the binary LSIRM above with the w_{ir} parameters equated for dummy coded variables that correspond to the same original items. In the resulting model, the estimates correspond closely to that of a graded response IRT model, although the models are strictly not identical. The number of score levels can be different across items as long as the lowest score is coded 0 for all items.

Value

An object of class `LSMfit` with values

<code>theta</code>	θ_p estimates
<code>b</code>	b_i estimates
<code>z</code>	z_{pr} estimates
<code>w</code>	z_{ir} estimates
<code>logL</code>	value of the loglikelihood at convergence
<code>starts</code>	the starting values used
<code>as_starts</code>	a list containing the parameter estimates, suitable to be used as argument for <code>starts</code> in a new run
<code>internal</code>	various matrices used internally

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

References

- Bergner, Y., Halpin, P., & Vie, J. J. (2022). Multidimensional Item Response Theory in the Style of Collaborative Filtering. *Psychometrika*, **87**(1), 266-288. <https://doi.org/10.1007/s11336-021-09788-9>
- Chen, Y., Li, X., & Zhang, S. (2019). Joint maximum likelihood estimation for high-dimensional exploratory item factor analysis. *Psychometrika*, **84**(1), 124-146. <https://doi.org/10.1007/s11336-018-9646-5>
- Molenaar, D., & Jeon, M.J. (2026). Joint maximum likelihood estimation of latent space item response models. *Psychometrika*, **91**(1), 335-359. <https://doi.org/10.1017/psy.2025.10068>
- Tutz, G. (2020). On the structure of ordered latent trait models. *Journal of Mathematical Psychology*, **96**, 102346. <https://doi.org/10.1016/j.jmp.2020.102346>.

See Also

[LSMselect](#) for selecting the number of latent space dimensions using cross-validation. [LSMsim](#) for simulating data according to the LSIRM. [LSMrotate](#) for rotating item and person coordinates.

Examples

```
#
# only binary items
#

# data sim with 1000 subjects and 20 binary items
# according to 2 dimensional latent space model (R=2)
set.seed(1111)
N=1000
nit=20
ndim_z=2
dat_obj=LSMsim(N,nit,ndim_z)
X=dat_obj$X
zt=dat_obj$par$zt      # rotated true z, see ?LSMsim and ?LSMrotate
wt=dat_obj$par$wt      # rotated true w

#fit model
results=LSMfit(X,2)

#plot the parameter recovery results
oldpar=par(mfrow=c(2,2))

s_p=sign(cor(results$z,zt))      # to correct for sign switches in the plots
s_i=sign(cor(results$w,wt))

plot(s_p[1,1]*zt[,1],results$z[,1]); abline(0,1)
plot(s_p[2,2]*zt[,2],results$z[,2]); abline(0,1)
plot(s_i[1,1]*wt[,1],results$w[,1]); abline(0,1)
plot(s_i[2,2]*wt[,2],results$w[,2]); abline(0,1)

par(oldpar)

#
# mixed scale items
#

# data sim with 1000 subjects and 20 mixed scale items
# according to 2 dimensional latent space model (R=2)
set.seed(1111)
N=1000
nit=20
ndim_z=2
nc=rpois(nit,2)+2      # number of response categories
                        # (between 2 and 7 for this seed)
dat_obj=LSMsim(N,nit,ndim_z,nc=nc)
X=dat_obj$X
```

```

zt=dat_obj$par$zt      # rotated true z, see ?LSMsim and ?LSMrotate
wt=dat_obj$par$wt      # rotated true w

#fit model
results=LSMfit(X,2)

#plot the parameter recovery results
oldpar=par(mfrow=c(2,2))

s_p=sign(cor(results$z,zt))      # to correct for sign switches in the plots
s_i=sign(cor(results$w,wt))

plot(s_p[1,1]*zt[,1],results$z[,1]); abline(0,1)
plot(s_p[2,2]*zt[,2],results$z[,2]); abline(0,1)
plot(s_i[1,1]*wt[,1],results$w[,1]); abline(0,1)
plot(s_i[2,2]*wt[,2],results$w[,2]); abline(0,1)

par(oldpar)

```

LSMrotate	<i>Rotate the person and item latent space parameter matrices to an echelon structure</i>
-----------	---

Description

This function rotates the person and item latent space parameter matrices to an echelon structure.

Usage

```
LSMrotate(z, w, method = c("echelon", "varimax", "echelon_step"))
```

Arguments

<code>z</code>	The N by ndim_z matrix of person coordinates z_{pr} to be rotated.
<code>w</code>	The nit by ndim_z matrix of item coordinates w_{ir} to be rotated.
<code>method</code>	Character string indicating the rotation method: "echelon" (the default), "echelon_step", or "varimax". See Details .

Details

[LSMfit](#) constrains the matrix of item coordinates w_{ir} to an echelon structure in fitting the LSIRM to data. Therefore, to compare results to other results (e.g., obtained using MCMC) or to the true values used to generate the data, it is necessary to rotate those other results/values to the same echelon structure. This rotation can be performed using `LSMrotate` with `method="echelon"` (the default) or `method="echelon_step"`; both produce the same echelon structure, up to numerical precision, via different algorithms. Following Wansbeek & Meijer (2000), `method="echelon"` uses a Cholesky decomposition, which works for an arbitrary number of latent space dimensions R (except 1). `method="echelon_step"` instead performs the explicit rotation steps by determining the

angle of rotation as described in Molenaar and Jeon (submitted); this method is only implemented for 2 or 3 latent space dimensions. `method="varimax"` performs a standard varimax rotation (see [varimax](#)) instead, which does **not** produce the echelon structure that LSMfit needs for identification; it is intended for improving the interpretability of an already-identified latent space after estimation (similar to varimax rotation in exploratory factor analysis), not for matching results to LSMfit's own echelon-structured output.

Value

A list containing

<code>zt</code>	The rotated matrix of z_{pr} parameters.
<code>wt</code>	The rotated matrix of w_{ir} parameters.
<code>rotMat</code>	The rotation matrix.

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

References

Molenaar, D., & Jeon, M.J. (2026). Joint maximum likelihood estimation of latent space item response models. *Psychometrika*, **91**(1), 335-359. <https://doi.org/10.1017/psy.2025.10068>

Wansbeek, T. & Meijer, E. (2000). *Measurement Error and Latent Variables in Econometrics*. Amsterdam: North-Holland.

See Also

[LSMfit](#) to fit LSIRM models.

Examples

```
set.seed(1111)
N=1000
nit=20
ndim_z=2

#some true values not following the echelon structure
z=matrix(rnorm(N*ndim_z),N,ndim_z)
w=matrix(rnorm(nit*ndim_z),nit,ndim_z)

# simulate data using these true values
dat_obj=LSMsim(N,nit,ndim_z,z=z,w=w)
X=dat_obj$X

#fit model
results=LSMfit(X,2)

#plot the parameter recovery results using the *unrotated* true values
#spoiler: will look like nothing
```

```

oldpar=par(mfrow=c(2,2))

s_p=sign(cor(results$z,z))      # to correct for sign switches in the plots
s_i=sign(cor(results$w,w))

plot(s_p[1,1]*z[,1],results$z[,1]); abline(0,1)
plot(s_p[2,2]*z[,2],results$z[,2]); abline(0,1)
plot(s_i[1,1]*w[,1],results$w[,1]); abline(0,1)
plot(s_i[2,2]*w[,2],results$w[,2]); abline(0,1)

#plot the parameter recovery results using the *rotated* true values
#spoiler: will look better

zt=dat_obj$par$zt      # rotated true z, see ?LSMsim and ?LSMrotate
wt=dat_obj$par$wt      # rotated true w

s_p=sign(cor(results$z,zt))      # to correct for sign switches in the plots
s_i=sign(cor(results$w,wt))

plot(s_p[1,1]*zt[,1],results$z[,1]); abline(0,1)
plot(s_p[2,2]*zt[,2],results$z[,2]); abline(0,1)
plot(s_i[1,1]*wt[,1],results$w[,1]); abline(0,1)
plot(s_i[2,2]*wt[,2],results$w[,2]); abline(0,1)

par(oldpar)

```

LSMselect

Selecting the Latent Space Dimensionality using K-fold Cross-Validation

Description

This function perform a K-fold cross validation to select the number of dimensions, R, of the latent space in the Latent Space Item Response Model (LSIRM). Model performance is evaluated using metrics based on the out-of-sample prediction accuracy, the area under the ROC curve, and the mean squared error.

Usage

```

LSMselect(X, maxDims=3, nfolds=5, penalty=NULL, C=NULL,
          starts=NULL, tol=.1, silent=TRUE)

```

Arguments

X A matrix of size N by n containing the binary or ordinal item scores, where N is the number of subjects and n is the number of items. The number of item score categories can be different across items as long as the lowest score is coded 0 for all items. NA's are allowed.

maxDims	The maximum number of dimensions R for the latent space to be considered. Should be at least 1 so that at least R=0 and R=1 are considered.
nfolds	The number of folds K.
penalty	The weight for the L2 penalty of pJML. If penalty is NULL (the default), a pJML is used with a weight of 1 (i.e., standard normal prior on all parameters).
C	Not yet supported: LSMselect only cross-validates pJML models. C should be left at NULL (the default); specifying it (for cJML, see LSMfit) currently causes LSMselect to stop with an error.
starts	Either a list containing starting values for the model parameters, or a character string indicating the method of starting value calculation: "ml", "wls", "mds", or "random" (see LSMfit for what each of these does). starts is used only once, up front, to obtain a single starting configuration for the largest (maxDims-dimensional) model, which is then reused as a warm start for smaller models and for subsequent folds; LSMselect therefore does not support LSMfit 's "default" (the ml/wls/20-random-starts cascade) or a numeric number of random starts, since repeating a multi-start search for every fold and dimension would be prohibitively slow for cross-validation.
tol	Convergence criterion: Iterations stop if the difference in loglikelihoods between two subsequent iterations is smaller than this number. Default is .1.
silent	Logical. If FALSE, iterations details are printed to the screen during estimation.

Details

LSMselect assigns the non-missing elements of the N by n matrix X randomly to one of the K folds making sure that the folds are (close to) equally sized. Then, maxDims+1 models (i.e., R=0, R=1, ..., R=maxDims) are fit leaving out the data of the first fold. Next, using the parameter estimates for each of the models, the data in the first fold are predicted. Using these predictions and the actual observations in the fold, the three metrics below are calculated. This scheme is repeated for all folds, so that each fold is held out of estimation once.

The metrics calculated are respectively based on the well known prediction accuracy, area under the ROC curve, and mean squared error. However, the metrics are unnormalized and -for accuracy and the ROC curve- are complements so that for all metrics lower values indicate a better model fit. This results in the following metrics:

Unnormalized Classification Error (UCE) The UCE is the number of incorrectly predicted item scores summed over folds

Unnormalized ROC Error (URE) The URE is the complement of the area under the ROC curve multiplied by the fold size and summed over folds

Residual Sum of Squares (RSS) The RSS is the sum of the squared residuals over folds For more details see Molenaar and Jeon (submitted).

Value

An object of class LSMselect with values

tot_metrics	The overall metrics (summed over folds)
fold_metrics	A list with separate entries for each metric containing the results for each fold separately

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

References

Molenaar, D., & Jeon, M.J. (2026). Joint maximum likelihood estimation of latent space item response models. *Psychometrika*, **91**(1), 335-359. <https://doi.org/10.1017/psy.2025.10068>

See Also

[LSMfit](#) for fitting LSIRM models and for details about the model.

Examples

```
# Toy example: compare between R=0 and R=1 for data that follows one dimensional
# latent space model (R=1) using only 2 folds

set.seed(1111)
N=1000
nit=20
ndim_z=1
dat_obj=LSMsim(N,nit,ndim_z)
X=dat_obj$X
LSMselect(X,1,nfolds=2)
```

LSMsim

Simulating Data according to the Latent Space Item Response Model

Description

This function simulates data according to the Latent Space Item Response Model (LSIRM) with an R dimensional latent space and binary and/or ordinal item scores.

Usage

```
LSMsim(N, nit, ndim_z, nc=NULL, theta=NULL, b=NULL, z=NULL, w=NULL, gamma=NULL)
```

Arguments

N	Sample size
nit	Number of items
ndim_z	Number of dimensions of the latent space, R.
nc	Vector of length nit containing the number of response categories for each item. If NULL (the default) all items are simulated to be binary
theta	N-dimensional vector of true person intercepts θ_p , if NULL these are drawn from a standard normal distribution

b	nit-dimensional vector of true item intercepts b_p , if NULL these are drawn from a uniform distribution
z	N by ndim_z matrix of true latent space person coordinates z_{pr} , if NULL these are drawn from a standard normal distribution
w	nit by ndim_z matrix of true latent space item coordinates w_{ir} , if NULL these are drawn from a standard normal distribution
gamma	a weight parameter for the Euclidean distances (see details), if NULL gamma=1

Details

Data is simulated according to the original LSIRM by Jeon et al. (2021):

$$\text{logit}(E(X_{pi})) = \theta_p + b_i - \gamma(\sum_{r=1}^R (z_{pr} - w_{ir})^2)^{1/2}$$

In [LSMfit](#), γ is fixed to one as it is not identified in a joint maximum likelihood framework (see Molenaar & Jeon, submitted). However, for data simulation, γ can be used to change the strength of the effect of z_{pr} and w_{ir} .

Value

A list containing:

X	the simulated data
par	a list containing the true parameter values, including wt and zt, the rotated matrices of w_{ir} and z_{pr} parameters.

Author(s)

Dylan Molenaar <d.molenaar@uva.nl>

References

Jeon, M., Jin, I. H., Schweinberger, M., & Baugh, S. (2021). Mapping unobserved item–respondent interactions: A latent space item response model with interaction map. *Psychometrika*, 86(2), 378–403. doi:10.1007/s11336-021-09762-5

Molenaar, D., & Jeon, M.J. (2026). Joint maximum likelihood estimation of latent space item response models. *Psychometrika*, **91**(1), 335–359. <https://doi.org/10.1017/psy.2025.10068>

See Also

[LSMfit](#) for fitting LSIRM models using joint maximum likelihood.

Examples

```
# data sim with 1000 subjects and 20 items according to 2 dimensional
# latent space model (R=2) with both binary and ordinal items
set.seed(1111)
N=1000
nit=20
ndim_z=2
```

```
nc=sample(c(2,3,5),nit,replace=TRUE)    # mix of 2, 3, and 5 point scales
dat_obj=LSMsim(N,nit,ndim_z,nc=nc)
X=dat_obj$X
zt=dat_obj$par$zt    # rotated z
wt=dat_obj$par$wt    # rotated w

#fit model
## Not run:
  results=LSMfit(X,2)

#plot the parameter recovery results
oldpar=par(mfrow=c(2,2))

s_p=sign(cor(results$z,zt))    # to correct for sign switches in the plots
s_i=sign(cor(results$w,wt))

plot(s_p[1,1]*zt[,1],results$z[,1]); abline(0,1)
plot(s_p[2,2]*zt[,2],results$z[,2]); abline(0,1)
plot(s_i[1,1]*wt[,1],results$w[,1]); abline(0,1)
plot(s_i[2,2]*wt[,2],results$w[,2]); abline(0,1)
par(oldpar)

## End(Not run)
```

Index

* **models**

- LSMboot, [2](#)
- LSMdist, [4](#)
- LSMfit, [5](#)
- LSMrotate, [10](#)
- LSMselect, [12](#)
- LSMsim, [14](#)

* **multivariate**

- LSMboot, [2](#)
- LSMdist, [4](#)
- LSMfit, [5](#)
- LSMrotate, [10](#)
- LSMselect, [12](#)
- LSMsim, [14](#)

- LSMboot, [2](#)
- LSMdist, [4](#)
- LSMfit, [2–5](#), [5](#), [10](#), [11](#), [13–15](#)
- LSMrotate, [9](#), [10](#)
- LSMselect, [9](#), [12](#)
- LSMsim, [9](#), [14](#)

- varimax, [11](#)