

Package ‘bifactor’

July 19, 2021

Type Package

Title Exploratory Factor, Bi-factor, Two-tier and Multiple-tier Modeling

Version 0.1.0

Date 2021-07-20

Description Fit exploratory factor, bi-factor, two-tier and multiple-tier models with correlated factors.

License GPL-3

LazyData TRUE

Depends R (>= 4.0)

Imports Rcpp, Rcsdp

URL <https://github.com/Marcosjnez/bifactor>

BugReports <https://github.com/Marcosjnez/bifactor/issues>

Encoding UTF-8

RoxygenNote 7.1.1

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Marcos Jimenez [aut, cre, cph],
Francisco J. Abad [aut, cph],
Eduardo Garcia-Garzon [aut, cph],
Luis E. Garrido [aut, cph]

Maintainer Marcos Jimenez <marcosjnezhquez@gmail.com>

R topics documented:

efast	2
get_target	4
random_oblq	5
random_orth	5
random_poblq	6
retr_oblq	7
retr_orth	7

retr_poblq	8
rotate	9
sim_twoTier	10
sl	12
twoTier	13

Index	16
--------------	-----------

efast	<i>Fast exploratory factor analysis.</i>
-------	--

Description

Fast exploratory factor analysis.

Usage

```
efast(R, n_factors, method = "minres", rotation = "oblimin", projection = "oblq",
      Target = NULL, Weight = NULL, PhiTarget = NULL, PhiWeight = NULL,
      oblq_blocks = NULL, normalize = FALSE, gamma = 0, epsilon = 1e-02, k = 0, w = 1,
      random_starts = 1L, cores = 1L, init = NULL, efa_control = NULL, rot_control = NULL)
```

Arguments

R	Correlation matrix.
n_factors	Number of common factors to extract.
method	EFA fitting method: "ml" (maximum likelihood for multivariate normal variables), "minres" (minimum residuals), "pa" (principal axis) and "minrank" (minimum rank). Defaults to "minres".
rotation	Rotation criterion. Available rotations: "varimax", "cf" (Crawford-Ferguson), "oblimin", "geomin", "target", "xtarget" (extended target) and "none". Defaults to "oblimin".
projection	Projection method. Available projections: "orth" (orthogonal), "oblq" (oblique), "poblq" (partially oblique). Defaults to "oblq".
Target	Target matrix for the loadings. Defaults to NULL.
Weight	Weight matrix for the loadings. Defaults to NULL.
PhiTarget	Target matrix for the factor correlations. Defaults to NULL.
PhiWeight	Weight matrix for the factor correlations. Defaults to NULL.
oblq_blocks	Vector with the number of factors for each oblique block. E.g.: c(2, 4) means that there are two blocks of oblique factors: one block with 2 factors and another block with 4 factors. Everything else is orthogonal. Defaults to NULL.
normalize	Kaiser normalization. Defaults to FALSE.
gamma	γ parameter for the oblimin criterion. Defaults to 0 (quartimin).
epsilon	ϵ parameter for the geomin criterion. Defaults to 0.01.

<code>k</code>	k parameter for the Crawford-Ferguson family of rotation criteria. Defaults to 0.
<code>w</code>	w parameter for the extended target criterion ("xtarget"). Defaults to 1L.
<code>random_starts</code>	Number of rotations with different random starting values. The rotation with the smallest cost function value is returned. Defaults to 1L.
<code>cores</code>	Number of cores for parallel execution of random starts. Defaults to 1L.
<code>init</code>	Initial uniquenesses values for exploratory factor analysis estimation. Defaults to NULL.
<code>efa_control</code>	List of control parameters for efa fitting. Defaults to NULL.
<code>rot_control</code>	List of control parameters for the rotation algorithm. Defaults to NULL.

Details

If `efa.control = NULL`, then `list(maxit = 1e4)` is passed to `efa.control`. If `rot_control = NULL`, then `list(maxit = 1000, eps = 1e-05)` is passed to `rot_control`, where `eps` is the absolute tolerance. When the objective function does not make a larger improvement than `eps`, the algorithm is assumed to converge.

If `Target` is provided but not `Weight`, then `Weight = 1 - Target` by default, which means a partially specified target rotation is performed. The same applies for `PhiTarget` and `PhiWeight`.

If `init = NULL`, then the squared multiple correlations of each item with the remaining ones are used as initial values (These are known to be upper bounds).

If a Heywood case is encountered, then `method = "minrank"` is automatically applied to ensure positive uniquenesses.

Value

List of class `efast` with the following components:

<code>efa</code>	List containing the following objects: <ul style="list-style-type: none"> • <code>loadings</code> - Unrotated loadings. • <code>uniquenesses</code> - Vector of uniquenesses. • <code>Rhat</code> - Correlation matrix predicted by the model. • <code>residuals</code> - Residual correlation matrix. • <code>f</code> - Objective value at the minimum. • <code>Heywood</code> - TRUE if any Heywood case is encountered and FALSE otherwise. • <code>iterations</code> - Number of iterations for the L-BFGS-B algorithm to converge. • <code>convergence</code> - TRUE if the L-BFGS-B algorithm converged and FALSE otherwise. • <code>method</code> - Method used to fit the exploratory factor analysis.
<code>rotation</code>	List of class <code>rotation</code> . Only if the argument <code>rotation</code> is not "none". See <code>rotate</code> for the components.
<code>elapsed</code>	Total amount spent for execution (in nanoseconds).

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Examples

```
## Not run:
# Simulate data:
sim <- sim_twoTier(n_generals = 2, groups_per_general = 5, items_per_group = 6)
scores <- MASS::mvrnorm(1e3, rep(0, nrow(sim$R)), Sigma = sim$R)
s <- cor(scores)

# Fit efa:
efa <- efast(s, n_factors = 12, method = "minres", rotation = "oblimin",
projection = "oblq", gamma = 0, random_starts = 10L, cores = 1L)

## End(Not run)
```

get_target

Get a target from a loading matrix.

Description

Get a target for the loading matrix using a custom or empirical cut-off.

Usage

```
get_target(loadings, Phi = NULL, cutoff = 0)
```

Arguments

loadings	A matrix of loadings.
Phi	A correlation matrix among the factors. Defaults to NULL.
cutoff	The cut-off used to create the target matrix. Defaults to 0.

Details

If cutoff is not 0, loadings smaller than such a cut-off are fixed to 0. When cutoff = 0, an empirical cut-off is used for each column of the loading matrix. They are the mean of the one-lagged differences of the sorted squared normalized loadings. Then, the target is determined by fixing to 0 the squared normalized loadings smaller than such cut-offs.

Value

A target matrix.

References

Garcia-Garzon, E., Abad, F. J., & Garrido, L. E. (2019). Improving bi-factor exploratory modeling: Empirical target rotation based on loading differences. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, 15(2), 45–55. <https://doi.org/10.1027/1614-2241/a000163>

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

`random_oblq`*Generate random oblique matrices.*

Description

Generate random oblique matrices from a standard normal distribution.

Usage

```
random_oblq(p, q)
```

Arguments

<code>p</code>	Number of rows.
<code>q</code>	Number of columns. Should not be greater than <code>p</code> .

Value

An oblique matrix with normally distributed data.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

`random_orth`*Generate random orthogonal matrices.*

Description

Generate random orthogonal matrices from a standard normal distribution. First, a matrix of random standard normal variables is simulated and then, the Q factor from the QR decomposition is returned.

Usage

```
random_orth(p, q)
```

Arguments

p	Number of rows.
q	Number of columns. Should not be greater than p.

Value

An orthogonal matrix with normally distributed data.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

random_poblq	<i>Generate a random partially oblique matrix.</i>
--------------	--

Description

First, a matrix is simulated from a standard normal distribution. Second, the matrix is normalized and the Gram-Schmidt process is performed between the oblique blocks. Finally, the orthogonal blocks correspond to those columns of the Q matrix from the QR decomposition.

Usage

```
random_poblq(p, q, oblq_blocks)
```

Arguments

p	Number of rows.
q	Number of columns. Should not be greater than p.
oblq_blocks	A vector with the number of factors for each oblique block. E.g.: c(2, 4) means that there are two blocks of oblique factors: one with 2 factors and another with 4 factors. Everything else is orthogonal.

Value

A partially oblique matrix.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Examples

```
random_poblq(p = 7, q = 7, oblq_blocks = c(3, 2))
```

retr_oblq	<i>Retraction of a matrix onto the oblique manifold.</i>
-----------	--

Description

Transform a matrix into an oblique matrix.

Usage

```
retr_oblq(X)
```

Arguments

X	A matrix.
---	-----------

Value

An oblique matrix.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

retr_orth	<i>Retraction of a matrix onto the orthogonal manifold.</i>
-----------	---

Description

Transform a matrix into an orthogonal matrix.

Usage

```
retr_orth(X)
```

Arguments

X	A matrix.
---	-----------

Value

An orthogonal matrix.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

retr_poblq	<i>Retraction of a matrix onto the partially oblique manifold. @description Transform a matrix into a partially oblique matrix.</i>
------------	---

Description

Retraction of a matrix onto the partially oblique manifold.

@description

Transform a matrix into a partially oblique matrix.

Usage

```
retr_poblq(X, oblq_blocks)
```

Arguments

X	A matrix.
oblq_blocks	A vector with the number of factors for each oblique block. E.g.: c(2, 4) means that there are two blocks of oblique factors: one with 2 factors and another with 4 factors. Everything else is orthogonal.

Value

A partially oblique matrix.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Examples

```
X <- replicate(8, rnorm(8))
retr_poblq(X, c(2, 3, 3))
```


rotate

*Fast rotation algorithm for factor analysis.***Description**

Riemannian Newton Trust-Region algorithm to quickly perform (parallel) rotations with different random starting values.

Usage

```
rotate(loadings, rotation = "oblimin", projection = "oblique",
       Target = NULL, Weight = NULL, PhiTarget = NULL, PhiWeight = NULL,
       oblique_blocks = NULL, gamma = 0, epsilon = 0.01, k = 0, w = 1,
       random_starts = 1L, cores = 1L, rot_control = NULL)
```

Arguments

loadings	Unrotated loading matrix.
rotation	Rotation criterion. Available rotations: "varimax", "cf" (Crawford-Ferguson), "oblimin", "geomin", "target", "xtarget" (extended target) and "none". Defaults to "oblimin".
projection	Projection method. Available projections: "orth" (orthogonal), "oblique" (oblique), "poblique" (partially oblique). Defaults to "oblique".
Target	Target matrix for the loadings. Defaults to NULL.
Weight	Weight matrix for the loadings. Defaults to NULL.
PhiTarget	Target matrix for the factor correlations. Defaults to NULL.
PhiWeight	Weight matrix for the factor correlations. Defaults to NULL.
oblique_blocks	Vector with the number of factors for each oblique block. E.g.: c(2, 4) means that there are two blocks of oblique factors: one block with 2 factors and another block with 4 factors. Everything else is orthogonal. Defaults to NULL.
gamma	γ parameter for the oblimin criterion. Defaults to 0 (quartimin).
epsilon	ϵ parameter for the geomin criterion. Defaults to 0.01.
k	k parameter for the Crawford-Ferguson family of rotation criteria. Defaults to 0.
w	w parameter for the extended target criterion ("xtarget"). Defaults to 1.
random_starts	Number of rotations with different random starting values. The rotation with the smallest cost function value is returned. Defaults to 1L.
cores	Number of cores for parallel execution of random starts. Defaults to 1L.
rot_control	List of control parameters for the rotation algorithm. Defaults to NULL.

Details

If `rot_control = NULL`, then `list(maxit = 1000,eps = 1e-05)` is passed to `rot_control`, where `eps` is the absolute tolerance. When the objective function does not make a larger improvement than `eps`, the algorithm is assumed to converge. If `Target` is provided but not `Weight`, then `Weight = 1 -Target` by default, which means a partially specified target rotation is performed. The same applies for `PhiTarget` and `PhiWeight`.

Value

List of class rotation with the following components:

loadings	Rotated loading matrix.
Phi	Correlation matrix among the factors.
T	Rotation matrix.
f	Objective value at the minimum.
iterations	Number of iterations for the rotation algorithm to converge.
convergence	TRUE if the algorithm converged and FALSE otherwise.
elapsed	Total amount of time spent for execution (in nanoseconds).

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Zhang, G., Hattori, M., Trichtinger, L. A., & Wang, X. (2019). Target rotation with both factor loadings and factor correlations. *Psychological Methods*, 24(3), 390–402. <https://doi.org/10.1037/met0000198>

sim_twoTier	<i>Simulate a bi-factor or two-tier population structure.</i>
-------------	---

Description

Simulate a bi-factor or two-tier population structure with cross-loading, pure items and correlated factors.

Usage

```
sim_twoTier(n_generals, groups_per_general, items_per_group,
loadings_g = "medium", loadings_s = "medium",
crossloadings = 0, pure = FALSE,
generals_rho = 0, groups_rho = 0)
```

Arguments

n_generals	Number of general factors.
groups_per_general	Number of group factors per general factor.
items_per_group	Number of items per group factor.
loadings_g	Loadings' magnitude on the general factors: "low", "medium" or "high". Defaults to "medium".
loadings_s	Loadings' magnitude on the group factors: "low", "medium" or "high". Defaults to "medium".
crossloadings	Magnitude of the cross-loadings among the group factors. Defaults to 0.
pure	Pure items on the general factors. Defaults to FALSE.
generals_rho	Correlation among the general factors. Defaults to 0.
groups_rho	Correlation among the group factors. Defaults to 0.

Details

sim_twoTier generates bi-factor and two-tier patterns with cross-loadings, pure items and correlations among the general and group factors. When crossloading is different than 0, one cross-loading is introduced for an item pertaining to each group factor. When pure is TRUE, one item loading of each group factor is removed so that the item loads entirely on the general factor. To maintain the item communalities constant upon these modifications, the item loading on the other factors may shrink (if adding cross-loadings) or increase (if setting pure items).

Loading magnitudes may range between 0.3-0.5 ("low"), 0.4-0.6 ("medium") and 0.5-0.7 ("high").

Value

List with the following objects:

lambda	Population loading matrix.
Phi	Population factor correlation matrix.
R	Population correlation matrix.
uniquenesses	Vector of population uniquenesses.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

sl	<i>Schmid-Leiman Transformation.</i>
----	--------------------------------------

Description

Schmid-Leiman transformation into a bi-factor or two-tier pattern.

Usage

```
sl(R, n_generals, n_groups, first_efa = NULL, second_efa = NULL)
```

Arguments

R	Correlation matrix.
n_generals	Number of general factors.
n_groups	Number of group factors.
first_efa	Arguments to pass to efast in the first-order factor extraction. See efast for the default arguments.
second_efa	Arguments to pass to efast in the second-order factor extraction. See efast for the default arguments.

Details

First, a hierarchical factor model is fitted using a second-order factor analysis on the factor correlation obtained from a first-order factor analysis. Then, the item loadings on the general factors are assumed to be the direct effects of the general factors according to such hierarchical model. On the other hand, the item loadings on the group factors become the originally first-order loadings post-multiplied by the diagonal matrix containing the root of the item uniquenesses.

Obviously, the first-order factor analysis should be oblique to perform a second exploratory factor analysis.

If the second-order solution does not use an orthogonal projection, then the correlation matrix among the general factors for the Schmid-Leiman solution is simply that obtained from such second-order solution.

Value

loadings	Loading matrix of the Schmid-Leiman solution.
first_order_solution	Object of class efast with the first-order solution.
second_order_solution	Object of class efast with the second-order solution.
uniquenesses	Vector of uniquenesses.
Rhat	Correlation matrix predicted by the (hierarchical) model.

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Examples

```
## Not run:
# Simulate data:
sim <- sim_twoTier(n_generals = 2, groups_per_general = 3, items_per_group = 5)
lambda <- sim$lambda
Target <- ifelse(lambda > 0, 1, 0)

# Target rotation for the first-order efa and oblimin for the second-order efa:
first <- list(rotation = "target", projection = "oblq", Target = Target)
second <- list(rotation = "oblimin", projection = "oblq", gamma = 0)

SL <- sl(sim$R, n_generals = 2, n_groups = 6, first, second)

## End(Not run)
```

twoTier

Fit an exploratory bi-factor, two-tier or multiple-tier model.

Description

Fit an exploratory bi-factor, two-tier or multiple-tier model with correlated factors.

Usage

```
twoTier(R, n_generals, n_groups, twoTier_method = "GSLiD", projection = "oblq",
PhiTarget = NULL, PhiWeight = NULL, oblq_blocks = NULL,
init_Target = NULL, method = "minres", maxit = 20L,
cutoff = 0, w = 1, random_starts = 1L, cores = 1L, init = NULL,
efa_control = NULL, rot_control = NULL,
SL_first_efa = NULL, SL_second_efa = NULL, verbose = TRUE)
```

Arguments

R	Correlation matrix.
n_generals	Number of general factors to extract.
n_groups	Number of group factors to extract.
twoTier_method	"GSLiD" and "SL" (Schmid-Leiman) Defaults to "GSLiD".
projection	Projection method. Available projections: "orth" (orthogonal), "oblq" (oblique) and "poblq" (partially oblique). Defaults to "oblq".
PhiTarget	Target matrix for the factor correlations. Defaults to NULL.

PhiWeight	Weight matrix for the factor correlations. Defaults to NULL.
oblq_blocks	Vector with the number of factors for each oblique block. E.g.: c(2, 4) means that there are two blocks of oblique factors: one block with 2 factors and another block with 4 factors. Everything else is orthogonal. Defaults to NULL.
init_Target	Initial target matrix for the loadings. Defaults to NULL.
method	EFA fitting method: "ml" (maximum likelihood for multivariate normal variable), "minres" (minimum residuals), "pa" (principal axis) or "minrank" (minimum rank). Defaults to "minres".
maxit	Maximum number of iterations for the GSLiD algorithm. Defaults to 20L.
cutoff	Cut-off used to update the target matrix upon each iteration. Defaults to 0.
w	w parameter for the extended target criterion ("xtarget"). Defaults to 1L.
random_starts	Number of rotations with different random starting values. The rotation with the smallest cost function value is returned. Defaults to 1L.
cores	Number of cores for parallel execution of multiple rotations. Defaults to 1L.
init	Initial uniquenesses values for exploratory factor analysis estimation. Defaults to NULL.
efa_control	List of control parameters for efa fitting. Defaults to NULL.
rot_control	List of control parameters for the rotation algorithm. Defaults to NULL.
SL_first_efa	List of arguments to pass to efast to perform the first-order solution for the Schmid-Leiman method. Defaults to NULL.
SL_second_efa	List of arguments to pass to efast to perform the second-order solution for the Schmid-Leiman method. Defaults to NULL.
verbose	Print the convergence progress information. Defaults to TRUE.

Details

If `efa.control = NULL`, then `list(maxit = 1e4)` is passed to `efa.control`. If `rot_control = NULL`, then `list(maxit = 1000, eps = 1e-05)` is passed to `rot_control`, where `eps` is the absolute tolerance. When the objective function does not make a larger improvement than `eps`, the algorithm is assumed to converge.

If `Target` is provided but not `Weight`, then `Weight = 1 - Target` by default, which means a partially specified target rotation is performed. The same applies for `PhiTarget` and `PhiWeight`.

If `init = NULL`, then the squared multiple correlations of each item with the remaining ones are used as initial values (These are known to be upper bounds).

If `init_Target` is provided, then an initial target by means of the Schmid-Leiman transformation is not necessary.

If `cutoff` is not 0, loadings smaller than such a cut-off are fixed to 0. When `cutoff = 0`, an empirical cut-off is used for each column of the loading matrix. They are the mean of the one-lagged differences of the sorted squared normalized loadings. Then, the target is determined by fixing to 0 the squared normalized loadings smaller than such cut-offs.

Value

List of class twoTier.

efa	List containing objects related to the exploratory factor analysis estimation. See efast.
twoTier	List with the following components: <ul style="list-style-type: none"> • loadings - Rotated loading matrix. • Phi - Factor correlation matrix. • T - Transformation matrix. • f - Objective value at the minimum. • iterations - Number of iterations performed by the rotation algorithm. • convergence - Convergence of the rotation algorithm. • uniquenesses - Vector of uniquenesses. • Rhat - Correlation matrix predicted by the model. • Target - Updated target matrix. • Weight - Weight matrix. It is the complement of the updated target. • GSLiD_iterations - Number of iterations performed by the GSLiD algorithm. • GSLiD_convergence - Convergence of the GSLiD algorithm. • min_congruences - Vector containing, for each iteration, the minimum Tucker's congruence between the current loading matrix and the previous loading matrix. • max_abs_diffs - Vector containing, for each iteration, the maximum absolute difference between the current loading matrix and the previous loading matrix.
elapsed	Total amount of time spent for execution (in nanoseconds).

References

Jiménez, M., Abad, F.J., Garcia-Garzon, E., Garrido, L.E. (2021, June 24). Exploratory Two-tier Modeling. Under review. Retrieved from https://osf.io/7aszj/?view_only=8f7bd98025104347a96f60a6736f5a64

Examples

```
## Not run: # Simulate data:
sim <- sim_twoTier(n_generals = 3, groups_per_general = 5, items_per_group = 6,
  generals_rho = 0.3)
scores <- MASS::mvrnorm(1e4, rep(0, nrow(sim$R)), Sigma = sim$R)
s <- cor(scores)

# Fit an exploratory two-tier model with GSLiD:
GSLiD <- twoTier(s, n_generals = 3, n_groups = 15, method = "minres",
  projection = "poblq", twoTier_method = "GSLiD", oblq_blocks = 3,
  random_starts = 10, cores = 8, w = 1, maxit = 20, verbose = TRUE)

## End(Not run)
```

Index

`efast`, [2](#)

`get_target`, [4](#)

`random_oblq`, [5](#)

`random_orth`, [5](#)

`random_poblq`, [6](#)

`retr_oblq`, [7](#)

`retr_orth`, [7](#)

`retr_poblq`, [8](#)

`rotate`, [9](#)

`sim_twoTier`, [10](#)

`sl`, [12](#)

`twoTier`, [13](#)