Package 'alsi'

March 4, 2026

Type Package

Title Aggregated Latent Space Index for Binary, Ordinal, and Continuous Data

Version 0.2.0

Date 2026-03-03

Description Provides three stability-validated pipelines for computing an Aggregated Latent Space Index (ALSI): a binary MCA pipeline (alsi_workflow()), an ordinal pipeline using homals alternating least squares optimal scaling (alsi_workflow_ordinal()), and a continuous ipsatized SVD pipeline (calsi_workflow()). All three pipelines share a common bootstrap dual-criterion stability framework (principal angles and Tucker congruence phi) for determining the number of dimensions to retain before index construction. The package is designed to complement Segmented Profile Analysis (SEPA) and is intended for psychometric scale construction and dimensional reduction in survey and clinical research.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports homals, stats, graphics, utils

Suggests paran, readxl, openxlsx, testthat (>= 3.0.0), knitr, rmarkdown, spelling

Depends R (>= 4.1.0)

Language en-US

NeedsCompilation no

Author Se-Kang Kim [aut, cre] (ORCID: <https://orcid.org/0000-0003-0928-3396>)

Maintainer Se-Kang Kim <se-kang.kim@bcm.edu>

Repository CRAN

Date/Publication 2026-03-04 08:40:18 UTC

Contents

alsi — *Compute Aggregated Latent Space Index (ALSI)*

Description

Calculates ALSI as a variance-weighted Euclidean norm of row principal coordinates within a retained K-dimensional MCA subspace.

Usage

```
alsi(Fmat, eig, K)
```

Arguments

`Fmat`	Matrix of row principal coordinates (N x K or larger)
`eig`	Vector of eigenvalues (inertias)
`K`	Integer, number of dimensions to aggregate

Value

S3 object of class `alsi` containing:

`alpha`	Numeric vector of ALSI values (length N), representing each individual's variance-weighted distance from the centroid in the retained MCA subspace

w Variance weights (length K), computed as the proportion of retained inertia for each dimension

alpha_vec Aggregated direction vector (length K), equal to sqrt(w), used for projecting category coordinates

K Number of dimensions used in aggregation

Examples

```
# Create example data
set.seed(123)
Fmat <- matrix(rnorm(100 * 4), nrow = 100, ncol = 4)
eig <- c(0.5, 0.3, 0.15, 0.05)

# Compute ALSI
a <- alsi(Fmat, eig, K = 3)
print(a)
hist(a$alpha, main = "Distribution of ALSI")
```

---

alsi_workflow *Complete ALSI Analysis Workflow*

Description

Runs a complete ALSI analysis including parallel analysis for dimensionality assessment, bootstrap stability evaluation, ALSI computation, and visualization.

Usage

```
alsi_workflow(
  data,
  vars,
  B_pa = 2000,
  B_boot = 2000,
  q = 0.95,
  seed = 20260123
)
```

Arguments

data Data frame or path to .xlsx file

vars Character vector of binary variable names

B_pa Number of permutations for parallel analysis (default: 2000)

B_boot Number of bootstrap resamples (default: 2000)

q Quantile for parallel analysis (default: 0.95)

seed Random seed for reproducibility

Value

List (returned invisibly) containing all analysis objects:

`pa`	Parallel analysis results (class `mca_pa`)
`boot`	Bootstrap stability results (class `mca_bootstrap`)
`fit`	MCA fit object (class `mca_fit`)
`alsi`	ALSI values (class `alsi`)
`K`	Number of dimensions retained based on parallel analysis

Examples

```
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")
results <- alsi_workflow(
  data   = ANR2,
  vars   = vars,
  B_pa   = 100,
  B_boot = 100
)
results$pa
results$boot
results$alsi
```

`alsi_workflow_ordinal` *Ordinal ALSI pipeline via homals ALS optimal scaling*

Description

Runs the four-stage ordinal ALSI pipeline:

1. Permutation parallel analysis (column-wise shuffle preserves marginals, destroys inter-item structure) determines K_PA.
2. Reference homals fit followed by varimax rotation on the stacked category score matrix (the loading analogue in homogeneity analysis). The same rotation matrix is applied to person scores.
3. Bootstrap dual-criterion stability. For each resample, homals is refitted and the category score matrix is Procrustes-aligned to the reference. Principal angle and Tucker congruence phi are computed on the *same* post-Procrustes matrix. K* is the largest k where ALL dimensions 1..k satisfy BOTH criteria simultaneously.
4. Eigenvalue-weighted linear ALSI index from K* retained rotated person scores (result can be negative; z-standardized version also returned).

Usage

```
alsi_workflow_ordinal(
  data,
  items,
  reversed_items = character(0L),
  scale_min = 1L,
  scale_max = 5L,
  n_permutations = 100L,
  pa_percentile = 95,
  B_boot = 1000L,
  angle_threshold_deg = 20,
  tucker_threshold = 0.85,
  seed = 12345L,
  itermax = 1000L,
  verbose = TRUE
)
```

Arguments

`data` A `data.frame` containing item columns.

`items` Character vector of item column names.

`reversed_items` Character vector of items to reverse-score ($x' = \text{scale\_min} + \text{scale\_max} - x$) before analysis.

`scale_min` Integer. Lowest valid response value (default 1).

`scale_max` Integer. Highest valid response value (default 5).

`n_permutations` Integer. Permutation replicates for Stage 1 (100).

`pa_percentile` Numeric. Null-distribution percentile cutoff (95).

`B_boot` Integer. Bootstrap replicates for Stage 3 (1000).

`angle_threshold_deg`
Numeric. Max principal angle in degrees for a dimension to pass the stability criterion (default 20).

`tucker_threshold`
Numeric. Min Tucker congruence phi for a dimension to pass the replicability criterion (default 0.85).

`seed` Integer. Random seed (default 12345).

`itermax` Integer. Max ALS iterations passed to homals (1000).

`verbose` Logical. Print progress messages (default TRUE).

Value

An S3 object of class `"alsi_ordinal"` with components:

ALSI_index Numeric vector (n). Raw eigenvalue-weighted linear composite. Can be negative.

ALSI_z Numeric vector (n). Z-standardized ALSI.

K_PA Integer. Dimensions retained by parallel analysis.

K_star Integer. Final model order after dual-criterion selection.

Z_ref Matrix n x K_PA. Varimax-rotated person scores.

C_ref Matrix P x K_PA. Varimax-rotated stacked category scores.

lambda_rot Numeric vector (K_PA). Eigenvalues (invariant to varimax rotation).

stability_table Data frame. Per-dimension stability metrics (eigenvalue, angle, Tucker phi, pass/fail, grade).

pa_table Data frame. Parallel analysis results per dimension.

n_skipped Integer. Bootstrap replicates discarded due to non-convergence or degenerate resamples.

call The matched call.

References

de Leeuw, J., & Mair, P. (2009). Gifi methods for optimal scaling in R: The package homals. *Journal of Statistical Software*, 31(4), 1-21.

Gifi, A. (1990). *Nonlinear multivariate analysis*. Wiley.

Lorenzo-Seva, U., & ten Berge, J. M. F. (2006). Tucker's congruence coefficient as a meaningful index of factor similarity. *Methodology*, 2, 57-64.

Takane, Y., Young, F. W., & de Leeuw, J. (1977). Nonmetric individual differences multidimensional scaling: An alternating least squares method with optimal scaling features. *Psychometrika*, 42, 7-67.

`ANR2` *ANR2: Binary Psychiatric Comorbidity Dataset*

Description

A binary indicator dataset recording the presence (1) or absence (0) of nine psychiatric diagnoses for a sample of patients. The dataset is included as the primary example dataset for the binary MCA pipeline (`alsi_workflow`).

Usage

```
data(ANR2)
```

Format

A data frame with 13 columns:

MDD Major Depressive Disorder (0/1)

DYS Dysthymia (0/1)

DEP Depressive disorder NOS (0/1)

PTSD Post-Traumatic Stress Disorder (0/1)

OCD Obsessive-Compulsive Disorder (0/1)

GAD Generalized Anxiety Disorder (0/1)

ANX Anxiety disorder NOS (0/1)

SOPH Social Phobia (0/1)

ADHD Attention Deficit Hyperactivity Disorder (0/1)

pre_EDI Pre-treatment EDI score (numeric)

post_EDI Post-treatment EDI score (numeric)

pre_bmi Pre-treatment BMI (numeric)

post_bmi Post-treatment BMI (numeric)

Examples

```
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")

results <- alsi_workflow(ANR2, vars = vars, B_pa = 100, B_boot = 100)
```

`calsi` *Compute Continuous Aggregated Latent Space Index (cALSI)*

Description

Calculates cALSI as a variance-weighted Euclidean norm of row coordinates within a retained K-dimensional ipsatized SVD subspace.

Usage

```
calsi(F, eig, K)
```

Arguments

`F` Matrix of row coordinates (N x K or larger)

`eig` Vector of eigenvalues

`K` Integer, number of dimensions to aggregate

Value

S3 object of class `calsi` containing:

`alpha` Numeric vector of cALSI values (length N)

`w` Variance weights (length K)

`alpha_vec` Aggregated direction vector (sqrt of weights)

`K` Number of dimensions used

calsi_vs_sepa_demo *Demonstrate what cALSI adds beyond SEPA*

Description

Demonstrate what cALSI adds beyond SEPA

Usage

```
calsi_vs_sepa_demo(data, K = 4, B_boot = 2000, seed = 20260206)
```

Arguments

data	Data matrix
K	Number of dimensions
B_boot	Bootstrap samples for stability
seed	Random seed

Value

List with comparison results

calsi_workflow *Complete cALSI Workflow for Continuous Data*

Description

Integrates parallel analysis, bootstrap stability, and cALSI computation.

Usage

```
calsi_workflow(
  data,
  B_pa = 2000,
  B_boot = 2000,
  q = 0.95,
  seed = 20260206,
  K_override = NULL
)
```

Arguments

`data`	Data frame or matrix of continuous variables
`B_pa`	Number of permutations for parallel analysis
`B_boot`	Number of bootstrap resamples
`q`	Quantile for parallel analysis
`seed`	Random seed
`K_override`	Optional: override parallel analysis K* with specified value

Value

List containing all analysis objects

`compare_sepa_calsi` *Compare SEPA plane-wise summaries with cALSI*

Description

Compare SEPA plane-wise summaries with cALSI

Usage

```
compare_sepa_calsi(fit, K, target_ids = NULL)
```

Arguments

`fit`	SVD fit object
`K`	Number of dimensions
`target_ids`	Optional vector of person IDs to highlight

Value

Data frame comparing SEPA and cALSI person-level indices

mca_align *Align MCA solution via Procrustes rotation with sign anchoring*

Description

Performs orthogonal Procrustes rotation to align a set of category coordinates to a reference solution, then applies sign correction to maximize agreement with the reference on each dimension.

Usage

```
mca_align(G, Gref)
```

Arguments

G	Matrix of category coordinates to align (M x K)
Gref	Reference matrix of category coordinates (M x K)

Value

List containing:

G_aligned	Matrix of aligned category coordinates (M x K), rotated and sign-corrected to match the reference
R	Orthogonal rotation matrix (K x K) used for alignment

Examples

```
# Create example matrices
set.seed(123)
Gref <- matrix(rnorm(20), nrow = 10, ncol = 2)
G <- Gref %*% matrix(c(0.8, 0.6, -0.6, 0.8), 2, 2)

# Align G to Gref
aligned <- mca_align(G, Gref)
print(aligned$G_aligned)
```

mca_bootstrap *Bootstrap-Based Subspace Stability Assessment*

Description

Evaluates reproducibility of retained MCA dimensions via bootstrap resampling. Quantifies stability using Procrustes principal angles (subspace-level) and Tucker's congruence coefficients (dimension-level).

Usage

```
mca_bootstrap(data, vars, K, B = 2000, seed = 20260123, verbose = TRUE)
```

Arguments

data	Data frame or path to .xlsx file
vars	Character vector of binary variable names
K	Integer, number of dimensions to retain and assess
B	Integer, number of bootstrap resamples (default: 2000)
seed	Integer, random seed for reproducibility
verbose	Logical, print progress messages

Value

S3 object of class mca_bootstrap containing:

ref	Reference MCA fit object (class mca_fit)
K	Number of dimensions assessed
B	Number of bootstrap resamples performed
angles	Matrix of principal angles in degrees (B x K), measuring subspace similarity between bootstrap and reference solutions
tucker	Matrix of Tucker congruence coefficients (B x K), measuring dimension-level similarity after Procrustes alignment
angles_summary	Summary statistics (median, 5th, 95th percentiles) for principal angles
tucker_summary	Summary statistics (median, 5th, 95th percentiles) for Tucker congruence coefficients

Examples

```
# Using included ANR2 dataset
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")
boot <- mca_bootstrap(ANR2, vars = vars, K = 3, B = 100)
print(boot)
```

mca_pa *Parallel Analysis for MCA Dimensionality Assessment*

Description

Compares observed MCA eigenvalues against reference distributions from permuted data to identify statistically meaningful dimensions.

Usage

```
mca_pa(
  data,
  vars,
  B = 2000,
  q = 0.95,
  seed = 20260123,
  max_dims = 20,
  verbose = TRUE
)
```

Arguments

`data`	Data frame or path to .xlsx file
`vars`	Character vector of binary variable names
`B`	Integer, number of permutations (default: 2000)
`q`	Numeric, reference quantile for retention (default: 0.95)
`seed`	Integer, random seed for reproducibility
`max_dims`	Integer, maximum dimensions to display in plot
`verbose`	Logical, print progress messages

Value

S3 object of class `mca_pa` containing:

`eig_obs`	Observed eigenvalues from the MCA of the original data
`eig_q`	Reference quantiles from permutation distribution
`eig_perm`	Matrix of permutation eigenvalues (B x dimensions)
`K_star`	Suggested number of dimensions to retain (where observed > reference)
`fit`	MCA fit object (class `mca_fit`) from original data
`q`	Quantile threshold used for comparison
`B`	Number of permutations performed

Examples

```
# Using included ANR2 dataset
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")
pa <- mca_pa(ANR2, vars = vars, B = 100)
print(pa$K_star)
```

`plot_category_projections` *Plot Category Projections in MCA Space*

Description

Visualizes category coordinates in a 2D MCA subspace and optionally displays projections onto the aggregated ALSI direction.

Usage

```
plot_category_projections(
  fit,
  K,
  alpha_vec = NULL,
  dim_pair = c(1, 2),
  cex = 0.8,
  top_n = 15
)
```

Arguments

`fit`	MCA fit object (class `mca_fit`)
`K`	Number of dimensions in retained subspace
`alpha_vec`	Optional aggregated direction vector (from `alsi()`)
`dim_pair`	Integer vector of length 2, dimensions to plot (default: c(1,2))
`cex`	Character expansion for labels
`top_n`	Number of top categories to display by projection (default: 15)

Value

No return value, called for side effects. The function creates a scatter plot of category coordinates in the specified 2D subspace, with category labels displayed. If `alpha_vec` is provided, it also prints the top categories ranked by their absolute projection onto the ALSI direction to the console.

Examples

```
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")
pa <- mca_pa(ANR2, vars = vars, B = 100, verbose = FALSE)
fit <- pa$fit
plot_category_projections(fit, K = pa$K_star)
```

`plot_domain_loadings` *Plot Domain Loadings in SVD Space*

Description

Visualizes domain loadings in a 2D subspace (biplot-style).

Usage

```
plot_domain_loadings(fit, dim_pair = c(1, 2), cex = 1)
```

Arguments

`fit`	SVD fit object (class `svd_fit`)
`dim_pair`	Integer vector of length 2, dimensions to plot
`cex`	Character expansion for labels

`plot_subspace_stability` *Plot Subspace Stability Diagnostics*

Description

Creates diagnostic plots showing distributions of principal angles and Tucker congruence coefficients across bootstrap resamples.

Usage

```
plot_subspace_stability(boot_obj)
```

Arguments

`boot_obj`	Object of class `mca_bootstrap`

Value

No return value, called for side effects. The function creates a two-panel figure with: (1) boxplots of principal angles (left panel), showing the distribution of subspace similarity across bootstrap resamples for each dimension; and (2) boxplots of Tucker congruence coefficients (right panel), showing dimension-level replicability with reference lines at phi = 0.85 (good) and phi = 0.95 (excellent).

Examples

```
data(ANR2)
vars <- c("MDD", "DYS", "DEP", "PTSD", "OCD", "GAD", "ANX", "SOPH", "ADHD")
boot <- mca_bootstrap(ANR2, vars = vars, K = 3, B = 100)
plot_subspace_stability(boot)
```

`plot_subspace_stability_cont` *Plot Subspace Stability Diagnostics for Continuous Data*

Description

Plot Subspace Stability Diagnostics for Continuous Data

Usage

```
plot_subspace_stability_cont(boot_obj)
```

Arguments

`boot_obj`	Object of class `svd_bootstrap`

`svd_align` *Align SVD solution via Procrustes rotation with sign anchoring*

Description

Align SVD solution via Procrustes rotation with sign anchoring

Usage

```
svd_align(B, Bref)
```

Arguments

`B`	Matrix of domain loadings to align
`Bref`	Reference matrix of domain loadings

Value

List with aligned coordinates and rotation matrix

svd_bootstrap *Bootstrap-Based Subspace Stability Assessment for Ipsatized SVD*

Description

Evaluates reproducibility of retained dimensions via bootstrap resampling. Uses Procrustes principal angles (subspace-level) and Tucker's congruence coefficients (dimension-level).

Usage

```
svd_bootstrap(data, K, B = 2000, seed = 20260206, verbose = TRUE)
```

Arguments

`data`	Data frame or matrix of continuous variables
`K`	Integer, number of dimensions to assess
`B`	Integer, number of bootstrap resamples (default: 2000)
`seed`	Integer, random seed for reproducibility
`verbose`	Logical, print progress messages

Value

S3 object of class `svd_bootstrap`

svd_pa *Parallel Analysis for Ipsatized SVD Dimensionality Assessment*

Description

Uses the paran package (Horn's parallel analysis with Longman-Allen-Chabassol bias adjustment) for dimensionality assessment, ensuring compatibility with SEPA methodology. Falls back to a built-in method if paran is unavailable.

Usage

```
svd_pa(data, B = 2000, q = 0.95, seed = 20260206, graph = TRUE, verbose = TRUE)
```

Arguments

`data` Data frame or matrix of continuous variables

`B` Integer, number of iterations for paran (default: 2000)

`q` Numeric, centile for retention threshold (default: 0.95)

`seed` Integer, random seed for reproducibility

`graph` Logical, whether to display the scree plot (default: TRUE)

`verbose` Logical, print progress messages

Details

This function primarily uses the paran package, which implements Horn's parallel analysis with the bias adjustment described in Longman, Cota, Holden, & Fekken (1989). This is the same method used in SEPA.

The paran package should be installed: install.packages("paran")

Value

S3 object of class `svd_pa` containing:

`eig_obs` Observed eigenvalues

`eig_adj` Adjusted eigenvalues (from paran)

`eig_rand` Random eigenvalues (threshold)

`K_star` Number of dimensions to retain

`fit` SVD fit object for downstream cALSI computation

`method` Method used ("paran" or "fallback")

Index