Package ‘SEPA’

March 26, 2026

Type Package

Title Segment Profile Extraction via Pattern Analysis

Version 0.1.0

Description Implements the Segment Profile Extraction via Pattern Analysis method for row-mean-centered multivariate data. Core capabilities include SVD-based row-isometric biplot construction, bias-corrected and accelerated, and percentile bootstrap confidence intervals for domain coordinates and per-person direction cosines, Procrustes alignment of bootstrap replicates across planes, parallel analysis for dimensionality selection, and segment profile reconstruction in planes defined by pairs of singular dimensions. A synthetic Woodcock-Johnson IV look-alike dataset is provided for examples and testing. The method is described in Kim and Grochowalski (2019) <doi:10.1007/s00357-018-9277-7>.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports boot (>= 1.3-28), parallel

Suggests writexl (>= 1.4.0), knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

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 2026-03-26 10:20:02 UTC

Contents

`boot_cis_all` *BCa (with percentile fallback) confidence intervals for all bootstrap indices*

Description

Loops over columns of a boot object and calls `boot.ci` for each, returning a tidy data frame. Falls back to percentile intervals if the BCa calculation fails.

Usage

```
boot_cis_all(boot_obj, type = c("bca", "perc"), level = 0.95, idx_vec = NULL)
```

Arguments

`boot_obj`	An object of class `"boot"` returned by `boot`.
`type`	Character vector passed to `boot.ci`'s `type` argument. Default `c("bca", "perc")`.
`level`	Numeric confidence level. Default `0.95`.
`idx_vec`	Integer vector of column indices to process. Defaults to all columns of `boot_obj$t`.

Value

A data frame with columns `index`, `lwr`, `upr`, and `method` (one row per element of `idx_vec`).

Examples

```
## Not run:
# See run_sepa() for an end-to-end example

## End(Not run)
```

`draw_sepa_biplot` *Draw a SEPA row-isometric SVD biplot*

Description

Produces a base-R row-isometric biplot for a specified pair of dimensions (`p1`, `p2`). All persons are plotted as grey dots; a subset specified by `ids_highlight` is overlaid in red and labelled. Domain loading vectors are drawn as arrows. The plot is optionally saved to a PDF.

Usage

```
draw_sepa_biplot(
  svd_fit,
  id_vec,
  domain_names,
  p1 = 1L,
  p2 = 2L,
  ids_highlight = NULL,
  out_file = NULL,
  a_scale = 35,
  t_scale = 40,
  arrow_col = "#1F4E79",
  hi_col = "red3",
  others_alpha = 0.3
)
```

Arguments

`svd_fit`	List with components `U` ($n \times K$ left singular vectors), `d` (length-K singular values), and `V` ($p \times K$ right singular vectors), as returned by `svd` or the format produced inside `run_sepa`.
`id_vec`	Vector of length n. Person IDs (used to match `ids_highlight`).
`domain_names`	Character vector of length p. Domain labels used for arrow annotations.
`p1`	Integer. First dimension (x-axis). Default `1L`.
`p2`	Integer. Second dimension (y-axis). Default `2L`.
`ids_highlight`	Optional vector of IDs to emphasise. Matched against `id_vec`. Default `NULL` (no highlighting).
`out_file`	Character or `NULL`. If a path is given the plot is also written to that PDF file. Default `NULL`.
`a_scale`	Numeric. Arrow scaling factor. Default `35`.
`t_scale`	Numeric. Label scaling factor. Default `40`.
`arrow_col`	Colour string for domain arrows and labels. Default `"#1F4E79"`.
`hi_col`	Colour string for highlighted persons. Default `"red3"`.
`others_alpha`	Alpha transparency for background persons. Default `0.30`.

Value

Invisibly returns a list with the plotting coordinates: `Fx`, `Fy` (person scores), `end_x`, `end_y` (arrow tips), `lab_x`, `lab_y` (domain labels).

Examples

```
X  <- as.matrix(fake_wj[, c("LT","ST","CP","AP","VP","CK","FR")])
Xs <- X - rowMeans(X)
sv <- svd(Xs)
draw_sepa_biplot(
  svd_fit      = list(U = sv$u, d = sv$d, V = sv$v),
  id_vec       = fake_wj$ID,
  domain_names  = c("LT","ST","CP","AP","VP","CK","FR"),
  p1 = 1L, p2 = 2L,
  ids_highlight = c(724, 944)
)
```

`fake_wj` *Synthetic Woodcock-Johnson IV look-alike dataset*

Description

A synthetic dataset generated by `simulate_sepa_fake_wj` that approximates the observed marginal distributions (means, SDs, and ranges) of seven WJ-IV broad ability scores while respecting the qualitative level-elevation / pattern-elevation structure assumed by SEPA. The original WJ-IV norming data are proprietary; this object provides a fully reproducible, publicly shareable substitute.

Usage

```
fake_wj
```

Format

A data frame with 5\,127 rows and 8 columns:

ID Integer person identifier (1–5127).

LT Long-term retrieval broad ability score.

ST Short-term working memory score.

CP Cognitive processing speed score.

AP Auditory processing score.

VP Visual processing score.

CK Comprehension-knowledge score.

FR Fluid reasoning score.

All domain scores are in a standard score metric (mean ≈ 100, SD ≈ 15) and clipped to the reported empirical range.

Three attributes capture the generative parameters: `B_loadings` (7×4 orthonormal loading matrix), `lambda` (PE dimension variances), and `sigma_LE` (LE SD).

Source

Generated by `simulate_sepa_fake_wj(n = 5127, seed = 20251127)`. See `data-raw/generate_fake_wj.R` for the exact code.

Examples

```
dim(fake_wj)
summary(fake_wj[, c("LT","ST","CP","AP","VP","CK","FR")])
```

`parallel_analysis_ipsatized` *Parallel analysis for ipsatized data*

Description

Determines the number of statistically significant singular dimensions in an ipsatized score matrix by comparing observed squared singular values to the `conf`-quantile of the null distribution obtained by column-permuting and re-ipsatizing the data `B` times.

Usage

```
parallel_analysis_ipsatized(
  Xstar,
  B = 2000L,
  Kmax = 10L,
  conf = 0.95,
  seed = 123L
)
```

Arguments

`Xstar`	Numeric matrix. Ipsatized (row-mean-centered) data, $n \times p$.
`B`	Integer. Number of permutation replicates. Default `2000`.
`Kmax`	Integer. Maximum number of dimensions to evaluate. Internally capped at $\min(n, p)$. Default `10`.
`conf`	Numeric in $(0, 1)$. Quantile of the null distribution used as threshold. Default `0.95`.
`seed`	Integer random seed. Default `123`.

Value

A named list with three elements:

`sig_dims` Integer vector of dimension indices (1-based) whose observed eigenvalue exceeds the null threshold.

`eig_obs` Numeric vector of length `Kmax`: observed squared singular values.

`thr` Numeric vector of length `Kmax`: permutation null thresholds at level `conf`.

Examples

```
X <- simulate_sepa_fake_wj(n = 300, seed = 1)
Xs <- X[, c("LT","ST","CP","AP","VP","CK","FR")]
Xs <- as.matrix(Xs) - rowMeans(as.matrix(Xs))   # ipsatize
pa <- parallel_analysis_ipsatized(Xs, B = 100, Kmax = 6, seed = 42)
pa$sig_dims
```

`percentile_ci_mat` *Percentile confidence intervals from a matrix of bootstrap draws*

Description

Percentile confidence intervals from a matrix of bootstrap draws

Usage

```
percentile_ci_mat(M, level = 0.95)
```

Arguments

`M`	Numeric matrix with bootstrap replicates in rows and statistics in columns.
`level`	Numeric confidence level. Default `0.95`.

Value

A two-column matrix with columns `qlo` and `qhi`, one row per column of `M`.

Examples

```
set.seed(1)
M <- matrix(rnorm(1000 * 5), 1000, 5)
percentile_ci_mat(M, level = 0.95)
```

`print.sepa_result` *Print method for sepa_result objects*

Description

Print method for sepa_result objects

Usage

```
## S3 method for class 'sepa_result'
print(x, ...)
```

Arguments

`x`	A `sepa_result` object.
`...`	Ignored.

Value

Invisibly returns `x`, the `sepa_result` object passed in. Called primarily for its side effect of printing a compact summary to the console, including sample size, number of domains, number of dimensions, parallel-analysis significant dimensions, and marker domains.

`run_sepa` *Run a complete SEPA analysis*

Description

Convenience wrapper that executes the full Subprofile Extraction via Pattern Analysis (SEPA) pipeline on a matrix of domain scores. The function ipsatizes the data, fits a rank-K row-isometric SVD biplot, computes SEPA statistics (plane-fit rho and direction cosines), runs parallel analysis, bootstraps domain coordinates with BCa confidence intervals, and bootstraps per-person cosines with percentile confidence intervals.

Usage

```
run_sepa(
  data,
  K = 4L,
  target_ids = NULL,
  B_dom = 2000L,
  B_cos = 2000L,
  alpha_ci = 0.95,
  seed = 20251003L,
  pa_B = 2000L,
  use_parallel = FALSE,
  ncores = NULL,
  run_pa = TRUE,
  run_boot_dom = TRUE,
  run_boot_cos = TRUE,
  verbose = TRUE
)
```

Arguments

`data`	A numeric matrix or data frame of domain scores. Rows are persons; columns are domains. An optional column named `"ID"` is used as the person identifier and removed before analysis.
`K`	Integer. Number of SVD dimensions to retain. Default `4L`.

`target_ids`	Optional vector of person IDs (matched against the `ID` column or row position) for which per-person exemplar tables are assembled. `NULL` disables exemplar output. Default `NULL`.
`B_dom`	Integer. Bootstrap replicates for domain-coordinate CIs. Default `2000L`.
`B_cos`	Integer. Bootstrap replicates for per-person cosine CIs. Default `2000L`.
`alpha_ci`	Numeric confidence level. Default `0.95`.
`seed`	Integer random seed. Default `20251003L`.
`pa_B`	Integer. Permutation replicates for parallel analysis. Default `2000L`.
`use_parallel`	Logical. Use parallel processing for the bootstrap? Default `FALSE`.
`ncores`	Integer or `NULL`. Number of cores. `NULL` uses `max(1, detectCores() - 1)`. Default `NULL`.
`run_pa`	Logical. Run parallel analysis? Default `TRUE`.
`run_boot_dom`	Logical. Run domain-coordinate bootstrap? Default `TRUE`.
`run_boot_cos`	Logical. Run per-person cosine bootstrap? Ignored unless `!is.null(target_ids)`. Default `TRUE`.
`verbose`	Logical. Print progress messages? Default `TRUE`.

Value

A named list of class `"sepa_result"` containing:

`call` The matched call.

`domains` Character vector of domain names.

`pid` Person ID vector.

`n`, `p`, `K` Dimensions used.

`ref_fit` List with `F` $(n \times K)$, `B` $(p \times K)$, `d` (singular values), `U`, `V` — the reference row-isometric SVD.

`Xstar` Ipsatized data matrix.

`sepa_stats` Output of `sepa_stats_all`: `rho`, `C_all`, `C_plane`.

`pa` Output of `parallel_analysis_ipsatized`, or `NULL`.

`boot_dom` Raw `boot` object for domain coordinates, or `NULL`.

`dom_coords` Data frame of domain coordinates with BCa CIs, or `NULL`.

`len2` Data frame of $||b_j||^2$ with BCa CIs and marker flag, or `NULL`.

`boot_cos` Raw `boot` object for per-person cosines, or `NULL`.

`cosine_tables` Named list of data frames (one per plane plus `"all"`) with point estimates and percentile CIs for the persons in `target_ids`, or `NULL`.

`dom_dom_cosines` List with `plane12` and `plane34` data frames of domain–domain cosines, or `NULL`.

`norms` Data frame with $||F_i^{(r)}||$ for exemplar persons, or `NULL`.

`rho_exemplar` Data frame with plane-fit rho for exemplar persons, or `NULL`.

Examples

```
res <- run_sepa(
  data         = fake_wj,
  K            = 4L,
  target_ids   = c(724, 944),
  B_dom        = 200L,
  B_cos        = 200L,
  seed         = 1L,
  pa_B         = 100L,
  run_boot_cos = TRUE,
  verbose      = TRUE
)
head(res$sepa_stats$rho)
res$pa$sig_dims
```

`sepa_stats_all` *Compute SEPA statistics: plane-fit rho and direction cosines*

Description

Given reference loading vectors `B_ref` and person score matrix `F_ref` from a row-isometric SVD biplot, computes for every person:

- the plane-fit correlation ρ in each plane,
- direction cosines with every domain in the full K-dimensional space,
- direction cosines within each plane.

Usage

```
sepa_stats_all(B_ref, F_ref, planes = list(c(1L, 2L), c(3L, 4L)), pid = NULL)
```

Arguments

`B_ref`	Numeric matrix $p \times K$. Domain loading vectors (right singular vectors from the ipsatized data SVD).
`F_ref`	Numeric matrix $n \times K$. Person score coordinates (left singular vectors scaled by singular values: $U\,\mathrm{diag}(d)$).
`planes`	List of integer vectors, each of length 2, specifying which pair of dimensions defines a plane. Default `list(c(1, 2), c(3, 4))`.
`pid`	Optional integer or character vector of length n providing person IDs. Defaults to `1:n`.

Value

A named list with three elements, each a tidy data frame:

`rho` Columns: `id`, `plane`, `rho`.

`C_all` Columns: `id`, `domain`, `C_all`. Direction cosines across all K dimensions.

`C_plane` Columns: `id`, `domain`, `C_plane`, `plane`. Per-plane cosines.

Examples

```
X  <- as.matrix(fake_wj[1:200, c("LT","ST","CP","AP","VP","CK","FR")])
Xs <- X - rowMeans(X)
sv <- svd(Xs)
B  <- sv$v[, 1:4]; F <- sv$u[, 1:4] %*% diag(sv$d[1:4])
rownames(B) <- c("LT","ST","CP","AP","VP","CK","FR")
res <- sepa_stats_all(B, F)
head(res$rho)
```

`simulate_sepa_fake_wj` *Simulate a synthetic Woodcock-Johnson IV look-alike dataset*

Description

Generates a data frame that approximates the observed marginal distributions (means, SDs, and ranges) of the seven WJ-IV broad ability scores while respecting the qualitative level-elevation (LE) / pattern-elevation (PE) structure assumed by SEPA. The data are produced from an additive model comprising a strong person-level elevation component (LE), a K-dimensional orthonormal pattern component (PE), and residual noise; columns are then linearly calibrated to the target statistics and clipped to the observed ranges. Because the original norming data are proprietary, this function provides a fully reproducible, publicly shareable substitute.

Usage

```
simulate_sepa_fake_wj(
  n = 5127L,
  domains = c("LT", "ST", "CP", "AP", "VP", "CK", "FR"),
  seed = 20251127L,
  K = 4L,
  sigma_LE = sqrt(0.25),
  lambda = c(0.3, 0.18, 0.11, 0.06),
  sigma_eps = sqrt(0.1),
  target = data.frame(domain = c("LT", "ST", "CP", "AP", "VP", "CK", "FR"), mean =
    c(100.2, 100.93, 99.64, 101.01, 100.79, 100.92, 99.99), sd = c(15.55, 15.72, 16.01,
    15.61, 15.91, 15.75, 15.58), min = c(37.04, 35.77, 12.26, 36.55, 31.76, 38.34,
    32.74), max = c(148.37, 159.3, 150, 151.35, 160.44, 153.93, 148.04), stringsAsFactors
    = FALSE),
  do_calibrate = TRUE,
  do_clip = TRUE
)
```

Arguments

`n`	Integer. Number of simulated cases. Default `5127`.
`domains`	Character vector of length 7. Domain abbreviations used as column names. Default `c("LT","ST","CP","AP","VP","CK","FR")`.
`seed`	Integer random seed passed to `set.seed`. Default `20251127`.
`K`	Integer. Number of orthogonal PE dimensions. Must be 4.
`sigma_LE`	Numeric. Standard deviation of the level-elevation component. Default `sqrt(0.25)`.
`lambda`	Numeric vector of length 4. PE dimension variances. Default `c(0.30, 0.18, 0.11, 0.06)`.
`sigma_eps`	Numeric. Residual noise SD. Default `sqrt(0.10)`.
`target`	Data frame with columns `domain`, `mean`, `sd`, `min`, `max` specifying the calibration targets for each domain. Defaults reproduce Table 2 of the associated paper.
`do_calibrate`	Logical. Linearly re-scale each column to match `target` mean and SD. Default `TRUE`.
`do_clip`	Logical. Clip each column to `[target$min, target$max]`. Default `TRUE`.

Value

A data frame with `n` rows and columns `ID`, `LT`, `ST`, `CP`, `AP`, `VP`, `CK`, `FR` (or as specified by `domains`). Three attributes are attached: `B_loadings` (the $p \times K$ orthonormal loading matrix), `lambda` (PE variances), and `sigma_LE`.

Examples

```
fake <- simulate_sepa_fake_wj(n = 200, seed = 1)
dim(fake)          # 200 x 8
colMeans(fake[, -1])
```

`write_long_to_wide` *Reshape a long data frame to wide and write a CSV*

Description

Pivots a three-column long data frame (id, time, value) to wide format and optionally prefixes the new column names.

Usage

```
write_long_to_wide(df, id_col, time_col, value_col, file, prefix = "")
```

Arguments

`df`	Data frame to pivot.
`id_col`	Name of the person-identifier column.
`time_col`	Name of the within-person variable column (e.g. domain).
`value_col`	Name of the value column.
`file`	Character path for the output CSV. Pass `NULL` or `""` to skip writing.
`prefix`	Optional prefix prepended to the new wide-format column names (empty string = no prefix).

Value

The wide data frame, invisibly.

Examples

```
long_df <- data.frame(
  id     = rep(1:3, each = 2),
  domain = rep(c("LT", "ST"), 3),
  value  = c(100, 105, 98, 110, 102, 107)
)
wide <- write_long_to_wide(long_df, "id", "domain", "value",
                           file = NULL)
wide
```

`write_matrix_wide` *Write an n x p matrix as a wide CSV with an ID column*

Description

Write an n x p matrix as a wide CSV with an ID column

Usage

```
write_matrix_wide(M, id, file, domain_names = NULL)
```

Arguments

`M`	Numeric matrix, $n \times p$.
`id`	Vector of length n providing row identifiers.
`file`	Character path for the output CSV. Pass `NULL` or `""` to skip writing.
`domain_names`	Optional character vector of length p. Column names for the domain columns. Defaults to `colnames(M)` or `"D1", "D2", ...` if those are absent.

Value

The data frame (ID + matrix columns), invisibly.

Examples

```
M   <- matrix(rnorm(6), nrow = 2)
out <- write_matrix_wide(M, id = c("A", "B"), file = NULL,
                         domain_names = c("X1","X2","X3"))
out
```

Index