Package ‘ssMRCD’

September 16, 2025

Type Package

Title Robust Estimators for Multi-Group and Spatial Data

Version 2.0.1

Maintainer Patricia Puchhammer <patricia.puchhammer@tuwien.ac.at>

Description Estimation of robust estimators for multi-group and spatial data including the casewise robust Spatially Smoothed Minimum Regularized Determinant (ssMRCD) estimator and its usage for local outlier detection as described in Puchhammer and Filzmoser (2023) <doi:10.1080/10618600.2023.2277875> as well as for sparse robust PCA for multi-source data described in Puchhammer, Wilms and Filzmoser (2024) <doi:10.48550/arXiv.2407.16299>. Moreover, a cellwise robust multi-group Gaussian mixture model (MG-GMM) is implemented as described in Puchhammer, Wilms and Filzmoser (2024) <doi:10.48550/arXiv.2504.02547>. Included are also complementary visualization and parameter tuning tools.

License GPL-3

Encoding UTF-8

LazyData true

Imports stats, graphics, robustbase, scales, ellipse, dbscan, ggplot2, expm, rrcov, DescTools, rootSolve, Matrix, Rcpp, RcppArmadillo, cellWise, methods, utils

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0), rnaturalearth, rnaturalearthdata, dplyr, tidyr, ggridges

Config/testthat/edition 3

Depends R (>= 4.0.0)

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Patricia Puchhammer [aut, cre, cph], Peter Filzmoser [aut]

Repository CRAN

Date/Publication 2025-09-16 11:00:02 UTC

Contents

`align` *Align Loadings of Principal Components*

Description

Aligns loadings per neighborhood for better visualization and comparison. Different options are available.

Usage

```
align(PC, type = "largest", vec = NULL)
```

Arguments

`PC` Array of loadings of size p x k x N as returned by `msPCA$PC`.

`type` Character string specifying the sign adjustment method. One of:

`"largest"` Sets the sign so that the largest (by absolute value) loading is positive, per neighborhood and component.
`"maxvar"` Makes the variable with the highest absolute value positive, per neighborhood and component.
`"nonzero"` Makes the loading with the largest distance from zero over neighborhoods positive, per neighborhood and comp
`"scalar"` Adjusts signs so that the scalar product with `vec` is positive, per neighborhood and component. `vec` can be a p-d
`"mean"` Like `"scalar"`, but `vec` is the mean of loadings across neighborhoods for each components.
`"none"` No sign adjustment; returns the raw `PC`.

vec — NULL or vector containing vectors for type "scalar". If vec is of dimension p × k, the same vec is used for all neighborhoods.

Value

Returns an array of loadings of size p times k times N.

Examples

```
set.seed(1)

# Note, that in this example the vectors are not feasible loadings.
COVS = list("a"= matrix(runif(16, -1, 1), 4), "b" = matrix(runif(16, -1, 1), 4))
COVS = lapply(COVS, function(x) x %*% t(x))

pca = msPCA(eta = 1, gamma = 0.5, COVS = COVS, k = 2)
x = pca$PC

align(PC = x, type = "largest")
align(PC = x, type = "maxvar")
align(PC = x, type = "mean")
align(PC = x, type = "nonzero")
align(PC = x, type = "scalar", vec = c(1,1,1,1))
```

biplot.msPCA *Biplot Visualization for msPCA Objects*

Description

Creates a biplot of the first two principal components from an msPCA object, displaying variables or groups in the component space.

Usage

```
## S3 method for class 'msPCA'
biplot(x, ...)
```

Arguments

x — An object of class msPCA.

... — Additional graphical parameters to customize the plot:

shape Integer or character; shape of the points (default: 43).

size Numeric; size of the points/text (default: 3).

alpha Numeric between 0 and 1; transparency of the points/text (default: 0.7).

color Character; determines the coloring scheme of points, either "variable" or "groups" (default: "variable").

Value

A `ggplot2` object representing the biplot of the first two principal components.

Examples

```
set.seed(236)
data <- matrix(rnorm(2000), ncol = 4)
groups <- sample(1:10, 500, replace = TRUE)
W <- time_weights(N = 10, c(3,2,1))
covs <- ssMRCD(data, groups = groups, weights = W, lambda = 0.3)
pca <- msPCA(eta = 0.3, gamma = 0.5,COVS = covs$MRCDcov, k = 2)
pca$PC = align(PC = pca$PC, type = "largest")
biplot(pca, alpha = 1, shape = 16, size = 5, color = "variable")
```

`cellMGGMM` *Calculation of the cellwise robust multi-group Gaussian mixture model*

Description

Performs robust estimation of multivariate location and scatter within predefined groups using an iiterative EM-based algorithm.

Usage

```
cellMGGMM(
  X,
  groups,
  alpha = 0.5,
  hperc = 0.75,
  nsteps = 100,
  crit = 1e-04,
  silent = TRUE,
  maxcond = 100
)
```

Arguments

`X` A numeric matrix or data frame with observations in rows and variables in columns.

`groups` A vector indicating the group membership for each observation (length must match ‘nrow(X)‘).

`alpha` A non-negative numeric value between ‘0.5‘ and ‘1‘ controlling the flexibility degree. Default is ‘0.5‘.

`hperc` A numeric value in ‘[0,1]‘ controlling robustness of the estimation. Default is ‘0.75‘.

`nsteps` Number of main iteration steps in the algorithm. Default is ‘100‘.

`crit` Convergence criterion for iterative updates. Default is ‘1e-4‘.

`silent` Logical; if ‘TRUE‘, suppresses progress output. Default is ‘FALSE‘.

`maxcond` Maximum allowed condition number for covariance matrices. Default is ‘100‘.

Value

A list containing:

`X` The original data matrix.

`Ximp` The imputed and/or scaled data matrix.

`groups` Vector specifying group assignments from the input.

`class` Vector indicating the most likely group membership for each observation, as inferred by the model.

`mu` A list of estimated location (mean) vectors for each group.

`Sigma` A list of estimated covariance matrices for each group.

`Sigmai` A list of estimated inverse covariance matrices for each group.

`probs` A matrix of class probabilities for each observation (rows = observations, columns = groups).

`pi_groups` A matrix of estimated mixture probabilities, where rows correspond to groups and columns to distributions.

`W` A binary matrix indicating outlying cells (0 = outlier, 1 = no outlier).

`Q` A matrix of penalty weights.

`Sigma_reg` A list of estimated target (regularization) matrices.

`rho` A vector of regularization factors used in the estimation.

`alpha` Flexibility parameter, as provided in the function input.

`hperc` A matrix or vector indicating the percentage of outlying cells per variable and group, based on input.

`nsteps` The number of iteration steps taken until convergence.

`objvals` The values of the objective function across the iteration steps.

References

Puchhammer, P., Wilms, I., & Filzmoser, P. (2025). A smooth multi-group Gaussian Mixture Model for cellwise robust covariance estimation. *ArXiv preprint* doi:10.48550/arXiv.2504.02547.

See Also

`residuals_mggmm`

Examples

```
data("weatherAUT2021")
cut_lon = c(min(weatherAUT2021$lon)-0.2, 12, 16, max(weatherAUT2021$lon) + 0.2)
cut_lat = c(min(weatherAUT2021$lat)-0.2, 48, max(weatherAUT2021$lat) + 0.2)
groups = groups_gridbased(weatherAUT2021$lon, weatherAUT2021$lat, cut_lon, cut_lat)
N = length(unique(groups))
model = cellMGGMM(X = weatherAUT2021[, c("p", "s", "vv", "t", "rsum", "rel")],
                  groups = groups,
                  alpha = 0.5)
```

geo_weights *Inverse Geographic Weight Matrix*

Description

Calculates a inverse-distance based weight matrix for the function ssMRCD (see details).

Usage

```
geo_weights(coordinates, groups)
```

Arguments

coordinates	matrix of coordinates of observations.
groups	vector of neighborhood groups.

Details

First, the centers (means of the coordinates given) c_i of each neighborhood is calculated. Then, the Euclidean distance between the centers is calculated and the weight is based on the inverse distance between two neighborhoods,

$$w_{ij} = \frac{1}{dist(c_i, c_j)}.$$

It is scaled according to a weight matrix.

Value

Returns a weighting matrix `W` and the coordinates of the centers per neighborhood `centersN`.

See Also

time_weights

Examples

```
coordinates = matrix(rnorm(1000), ncol = 2, nrow = 500)
groups = sample(1:5, 500, replace = TRUE)

geo_weights(coordinates, groups)
```

groups_gridbased *Creates Grid-Based Neighborhood Structure*

Description

This function creates a grid-based neighborhood structure for the ssMRCD function using cut-off values for two coordinate axis.

Usage

```
groups_gridbased(x, y, cutx, cuty)
```

Arguments

x	vector of first coordinate of data set.
y	vector of second coordinate of data set.
cutx	cut-offs for first coordinate.
cuty	cut-offs for second coordinate.

Value

Returns a neighborhood assignment vector for the coordinates x and y.

Examples

```
# get data
data(weatherAUT2021)

# set cut-off values
cut_lon = c(9:16, 18)
cut_lat = c(46, 47, 47.5, 48, 49)

# create neighborhood assignments
groups_gridbased(weatherAUT2021$lon,
                      weatherAUT2021$lat,
                      cut_lon,
                      cut_lat)
```

locOuts *Local Outlier Detection using Spatially Smoothed MRCD*

Description

Identifies local multivariate outliers using spatially smoothed robust covariance estimation (ssMRCD) as proposed by Puchhammer and Filzmoser (2023). For each observation, the Mahalanobis distance to its nearest neighbor is computed, and an adjusted boxplot is used to detect outliers.

Usage

```
locOuts(data, coords, groups, lambda, weights = NULL, k = NULL, dist = NULL)
```

Arguments

`data` A numeric matrix of observations (rows = observations, columns = variables).

`coords` A numeric matrix of spatial coordinates corresponding to the observations.

`groups` A vector assigning each observation to a neighborhood/group.

`lambda` Smoothing parameter for the `ssMRCD` estimator.

`weights` Optional weighting matrix for spatial smoothing. If omitted, inverse-distance weights are computed automatically.

`k` Integer. Number of nearest neighbors to use if `dist` is not provided.

`dist` Numeric. Use neighbors within this distance instead of `k`-nearest neighbors. If both are provided, `dist` is used.

Value

An object of class `"locOuts"` containing:

`outliers` Indices of detected outliers.

`next_distance` Vector of Mahalanobis next distances (min distance to neighbors).

`cutoff` Upper fence of the adjusted boxplot used as outlier threshold.

`coords` Matrix of observation coordinates.

`data` Original data matrix.

`groups` Group assignments.

`k, dist` Neighborhood comparison parameters used.

`centersN` Centers of neighborhoods.

`matneighbor` Binary matrix indicating which neighbors were used for each observation.

`ssMRCD` The fitted `ssMRCD` object.

References

Puchhammer, P. and Filzmoser, P. (2023). Spatially Smoothed Robust Covariance Estimation for Local Outlier Detection. *Journal of Computational and Graphical Statistics*, 33(3), 928–940. doi:10.1080/10618600.2023.2277875

See Also

ssMRCD, plot.locOuts

Examples

```
data <- matrix(rnorm(2000), ncol = 4)
coords <- matrix(runif(1000), ncol = 2)
groups <- sample(1:10, 500, replace = TRUE)
result <- locOuts(data, coords, groups, lambda = 0.3, k = 10)
```

msPCA *Compute Sparse Multi-Source Principal Components*

Description

Estimates sparse principal components from multiple covariance or correlation matrices using an ADMM-based optimization routine (see Puchhammer, Wilms and Filzmoser, 2024).

Usage

```
msPCA(
  eta,
  gamma,
  COVS,
  k = ncol(COVS[[1]]),
  adjust_eta = TRUE,
  convergence_plot = FALSE,
  n_max = 200,
  rho = list(NA, TRUE, 100, 1),
  eps = c(1e-05, 1e-04, 0.1, 50),
  show_progress = FALSE
)
```

Arguments

`eta` Numeric or numeric vector. Controls the overall sparsity level. If a single value is provided, it will be used directly. If a vector is given, the optimal value will be selected via internal model selection.

`gamma` Numeric or numeric vector. Controls the distribution of sparsity across components. If a single value is provided, the optimal `eta` is selected automatically.

`COVS` A list of covariance or correlation matrices (one per data source or group).

`k` Integer. Number of principal components to compute. If not specified, all components are estimated.

`adjust_eta` Logical. If `TRUE` (default), the sparsity parameter `eta` is adjusted based on the variance structure.

`convergence_plot`
Logical. If `TRUE`, a convergence diagnostic plot is displayed of either the residuals or the loading entries (default: `FALSE`).

`n_max` Integer. Maximum number of ADMM iterations (default: 200).

`rho` List of parameters controlling the ADMM penalty parameter `rho`, with the following elements:

1 Initial value for `rho` (default: `NA`).

2 Logical; whether to increase `rho` if convergence is not reached (default: `TRUE`).

3 Maximum value for `rho` (default: 100). You may need to increase this for high-dimensional problems.

4 Step size for increasing `rho` (default: 1).

`eps` Numeric vector of tolerance parameters used in optimization. Includes:

1 Tolerance for soft-thresholding (default: `1e-5`).

2 Tolerance for ADMM convergence (default: `1e-4`).

3 Tolerance for convergence of the internal root-finding step (default: `1e-1`).

4 Maximum number of iterations for the root finder (default: 50).

`show_progress` Logical. Indicates whether progress bars should be displayed.

Value

An object of class `"msPCA"` containing the following elements:

`PC` Array of dimension p x k x N of loading vectors.

`p` Number of variables.

`N` Number of neighborhoods.

`k` Number of components.

`COVS` List of covariance matrices sorted by neighborhood.

`gamma` Sparsity distribution.

`eta` Amount of sparsity.

`converged` Logical, if ADMM converged with given specifications.

`n_steps` Number of steps used.

`residuals` Primary and secondary residuals.

References

Puchhammer, P., Wilms, I., & Filzmoser, P. (2024). Sparse outlier-robust PCA for multi-source data. *ArXiv Preprint*. doi:10.48550/arXiv.2407.16299

See Also

ssMRCD, plot.msPCA, summary.msPCA, biplot.msPCA, screeplot.msPCA, scores, align

Examples

```
C1 = diag(c(1.1, 0.9, 0.6, 0.5, 2))
C2 = matrix(runif(25, -1, 1), 5, 5)
C2 = t(C2) %*% C2
C3 = matrix(runif(25, -1, 1), 5, 5)
C3 = t(C3) %*% C3

pca1 = msPCA(eta = 1, gamma = 0.5, COVS = list(C1, C2, C3), k = 3,
             n_max = 100, rho = list(NA, TRUE, 100, 1), show_progress = FALSE)
summary(pca1)

pca2 = msPCA(eta = seq(0, 3, 0.25), gamma = 1, COVS = list(C1, C2, C3), k = 3,
             n_max = 100, rho = list(NA, TRUE, 100, 1), show_progress = FALSE)
summary(pca2)
```

plot.locOuts *Diagnostic Plots for Local Outlier Detection (‘locOuts‘)*

Description

Produces diagnostic plots for local outlier detection results returned by locOuts. Available visualizations include a histogram of next distances, spatial distribution of next distances, and a parallel coordinate plot (PCP) for a selected observation and their neighborhood.

Usage

```
## S3 method for class 'locOuts'
plot(
  x,
  type = c("hist", "spatial", "pcp"),
  scale = c("none", "minmax", "zscore"),
  bins = 30,
  observation = 1,
  ...
)
```

Arguments

x An object of class "locOuts" obtained from locOuts.

type A character vector indicating which plots to generate. Options are:

"hist" Histogram of next distances with cutoff visualized.

"spatial" Spatial distribution of observations, colored by relative next distance.

"pcp" Parallel coordinate plot for an observation and its neighbors.

`scale` Character indicating how variables are scaled in the parallel coordinate plot. One of:

- **"none"** Use raw values (no scaling).
- **"minmax"** Min-max scaling to [0, 1].
- **"zscore"** Standardization: mean 0, standard deviation 1.

`bins` Integer, number of histogram bins (default = 30).

`observation` Integer or character; index or name of a specific observation to analyze in the PCP plot. Used only when `type` includes `"pcp"`.

`...` Additional parameters passed to low-level plotting functions (currently unused in ggplot versions).

Details

The function visualizes outlier behavior in different ways:

- **Histogram**: Shows the distribution of next distances across observations. The cutoff is shown as a dashed line.
- **Spatial Plot**: 2D plot of observation coordinates. Color encodes the ratio of next distance to cutoff.
- **Parallel Coordinate Plot (PCP)**: Shows scaled values across all variables for a selected observation (in red) and its neighbors (in blue or grey). The type of scaling can be controlled via the `scale` parameter.

Value

A named list with elements:

`p_hist` ggplot object of the histogram (or `NULL` if not requested).

`p_spatial` ggplot object of the spatial plot (or `NULL`).

`p_pcp` ggplot object of the parallel coordinate plot (or `NULL`).

See Also

`locOuts`

Examples

```
set.seed(1)
data <- matrix(rnorm(2000), ncol = 4)
coords <- matrix(rnorm(1000), ncol = 2)
groups <- sample(1:10, 500, replace = TRUE)
outs <- locOuts(data = data,
                coords = coords,
                groups = groups,
                lambda = 0.3,
                k = 10)

# Generate all plots
```

```
plots <- plot(outs,
              type = c("hist", "spatial", "pcp"),
              observation = outs$outliers[1],
              scale = "minmax")
plots$p_hist
plots$p_spatial
plots$p_pcp
```

plot.msPCA *Plot Method for msPCA Objects*

Description

Generic plotting interface for objects of class "msPCA". Depending on the type argument, this function visualizes loadings, scores, screeplots, biplots, or score distances.

Usage

```
## S3 method for class 'msPCA'
plot(x, type = c("loadings"), ...)
```

Arguments

x	An object of class "msPCA".
type	Type of plot to produce. One of: "loadings" (default), "screeplot", "scores", "biplot", or "score_distances".
...	Additional arguments passed to internal functions.

Details

If type = "loadings" a heatmap of loadings across groups is displayed. Optional input arguments are:

k	Integer. The k-th principal component will be plotted.
text	Boolean, whether the loading values should also be displayed as text. Default FALSE.
size	Numeric. Text size when text is TRUE.
gnames	Character vector. Names for groups (shown in plots).
cnames	Character vector. Names for variables (shown in plots).
textrotate	Rotation angle of text in the heatmap.
tolerance	Numeric, default 1e-04. Specifies the band of white color values around zero.

If type = "screeplot" boxplots of the explained variance per component and cumulative variance per group are plotted. Optional input arguments are:

text	Boolean, whether the loading values should also be displayed as text. Default TRUE.
size	Numeric. Text size when text is TRUE.

`gnames` Character vector. Names for groups (shown in plots).
`cutoff` Scalar with default value 0.8. The cumulative percentage cutoff value for the overall explained variance.
`textrotate` Rotation angle of text in the heatmap.

If type = `"biplot"` the loadings are visualized in the first and second component over all groups. Optional input arguments are:

`color` Character. Either `"variable"` (default) when the color should be connected to the variables or `"groups"` if the color
`size` Numeric. Text size when `text` is `TRUE`.
`alpha` Alpha value for the loading points, default is `0.7` .

If type = `"scores"` a histogram of the k-th scores per group are shown. Optional input arguments are:

`k`
`ssMRCD` An object of class `ssMRCD` including list elements `MRCDcov`, `MRCDmu`, `mX`. Alternatively, `X`, `groups`, `mu`, `Sigma` can be
Alternatively, a `ssMRCD` object can be given.
`alpha`

If type = `"score_distances"` a distance-distance plot of score and orthogonal distances is shown for each group. of the k-th scores per group are shown. Optional input arguments are:

`k` Integer. Using the first k PCs for the distances. Default is the number of provid
Alternatively, a `ssMRCD` object can be given.
`shape` Point shape.
`size` Numeric. Point size.
`alpha` Alpha value for the points, default is `0.7` .

Value

A ggplot2 object or list of plots, depending on the type.

See Also

`msPCA`, `align`, `screeplot.msPCA`, , `biplot.msPCA`, `scores`,

Examples

```
# set seed
set.seed(236)

# create data and setup
data = matrix(rnorm(2000), ncol = 4)
groups = sample(1:10, 500, replace = TRUE)
W = time_weights(N = 10, c(3,2,1))
```

```
# calculate covariances
covs = ssMRCD(data, groups = groups, weights = W, lambda = 0.3)

# calculate sparse PCA
pca = msPCA(eta = 1.3, gamma = 0.7, COVS = covs$MRCDcov)

# plot screeplot
plot(x = pca, type = "screeplot")

# align and plot loadings
pca$PC = align(PC = pca$PC, type = "mean")
plot(x = pca, type = "loadings", k = 1)
pca$PC = align(PC = pca$PC, type = "maxvar")
plot(x = pca, type = "loadings", k = 1)
pca$PC = align(PC = pca$PC, type = "largest")
plot(x = pca, type = "loadings", k = 1)

# plot different PCA plots
plot(x = pca, type = "score_distances", k = 2,
groups = groups, X = data, mu = covs$MRCDmu, Sigma = covs$MRCDcov)
plot(x = pca, type = "biplot", color = "variable")
plot(x = pca, type = "scores", ssMRCD = covs, k = 1)
plot(x = pca, type = "loadings", k = 1)
```

plot.ssMRCD *Plot Method for ssMRCD Object*

Description

Produces diagnostic plots for an object of class "ssMRCD" including convergence behavior and visualizations of the covariance matrices.

Usage

```
## S3 method for class 'ssMRCD'
plot(
  x,
  type = c("convergence", "ellipses", "ellipses_geo"),
  variables = NULL,
  geo_centers = NULL,
  manual_rescale = 1,
  tolerance = 0.95,
  ...
)
```

Arguments

x	An object of class "ssMRCD".

type Character string or vector specifying the type of plot(s) to produce. Available options are "convergence", "ellipses", and "ellipses_geo". See Details.

variables A character vector of length 2 specifying the variable names (columns of the data) used to compute and plot the covariance ellipses.

geo_centers A matrix specifying the spatial/geographical coordinates of the group centers. Required when type = "ellipses_geo".

manual_rescale Numeric scaling factor to adjust the size of ellipses in "ellipses_geo" plots.

tolerance Numeric value (between 0 and 1) specifying the quantile used to define tolerance ellipses. Default is 0.95.

... Further arguments passed to plotting functions.

Details

type = "convergence": Displays the convergence behavior of the objective function across C-step iterations for each initialization. Red lines indicate non-monotonic convergence.

type = "ellipses": Shows Mahalanobis tolerance ellipses for each group based on their estimated covariance matrix. Only the two variables specified in variables are visualized. The global MCD ellipse may be shown for comparison (if included elsewhere).

type = "ellipses_geo": Projects group-wise covariance ellipses onto a geographical coordinate system (e.g., spatial map), using the positions given in geo_centers. The ellipses are scaled using manual_rescale and drawn using the same two variables as for type = "ellipses".

Value

A named list of ggplot2 plot objects:

plot_convergence
Plot showing convergence diagnostics (if "convergence" selected).

plot_ellipses Plot of covariance ellipses in variable space (if "ellipses" selected).

plot_geoellipses
Plot of covariance ellipses in geographical space (if "ellipses_geo" selected).

See Also

ssMRCD, locOuts, plot.locOuts

Examples

```
set.seed(1)
data <- matrix(rnorm(2000), ncol = 4)
colnames(data) <- paste0("V", 1:4)
coords <- matrix(rnorm(1000), ncol = 2)
groups <- sample(1:10, 500, replace = TRUE)
lambda <- 0.3

outs <- locOuts(data = data,
                              coords = coords,
                              groups = groups,
                                    lambda = lambda,
                                    k = 10)

# Plot convergence
plot(x = outs$ssMRCD, type = "convergence")

# Plot ellipses in variable space
plot(x = outs$ssMRCD, type = "ellipses", variables = c("V1", "V2"))

# Plot ellipses in geographical space
centers <- matrix(rnorm(20), ncol = 2)  # example centers for 10 groups
plot(x = outs$ssMRCD, type = "ellipses_geo", geo_centers = centers, variables = c("V1", "V2"))
```

residuals.ssMRCD *Residual Method from an ssMRCD Object*

Description

Computes group-wise Mahalanobis residuals (standardized distances) using the robust local covariance and location estimates from an ssMRCD object. Residuals can be computed for the fitted data or for new data, and optionally summarized as a trimmed mean.

Usage

```
## S3 method for class 'ssMRCD'
residuals(object, ...)
```

Arguments

object	An object of class "ssMRCD", typically the result of ssMRCD.
...	Additional arguments, see Details.

Details

The function supports several modes of use, controlled by the type argument in ...:

type "residuals" (default), "trimmed_mean", or "additional_data".

X A numeric matrix of new observations to compute residuals for. Required if type = "additional_data".

groups A vector of group assignments for the new data in X. Required if type = "additional_data".

alpha A numeric value (default taken from the ssMRCD object if missing) indicating the quantile for trimmed mean calculation. Only used if type = "trimmed_mean".

Notes:

- If type = "residuals", residuals are computed for the original data stored in the ssMRCD object.

- If type = "additional_data", both X and groups must be provided. All residuals of X are returned (i.e., alpha = 1 is used internally).
- If type = "trimmed_mean", the mean of the alpha proportion of smallest residual norms is returned. This is also used for parameter tuning.

Value

Depending on the type:

"residuals" **or** "additional_data" A numeric matrix of residuals.

"trimmed_mean" A single numeric value: the trimmed mean of residual norms.

See Also

ssMRCD

Examples

```
# Create data
x1 <- matrix(runif(200), ncol = 2)
x2 <- matrix(rnorm(200), ncol = 2)
x <- list(x1, x2)

# Define neighborhood weights
W <- matrix(c(0, 1, 1, 0), ncol = 2)

# Compute ssMRCD
localCovs <- ssMRCD(x, weights = W, lambda = 0.5)

# Residuals for original data (all)
head(residuals(localCovs, type = "residuals"))

# Trimmed mean of residual norms
residuals(localCovs, type = "trimmed_mean", alpha = 0.8)

# Residuals for new data
newX <- matrix(rnorm(20), ncol = 2, nrow = 10)
newGroups <- rep(2, 10)
residuals(localCovs, type = "additional_data", X = newX, groups = newGroups)
```

residuals_mggmm *Calculation of Residuals for the Multi-Group GMM*

Description

This function calculates the cell-wise residuals for each observation based on the fitted parameters of a multi-group Gaussian Mixture Model (GMM) and the cellwise outlyingness pattern in matrix ‘W‘.

Usage

```
residuals_mggmm(X, groups, Sigma, mu, probs, W, set_to_zero = TRUE)
```

Arguments

`X`	A numeric data matrix or data frame with observations in rows and variables in columns.
`groups`	A vector indicating pre-defined group membership for each observation (length must match ‘nrow(X)‘).
`Sigma`	A list of estimated covariance matrices.
`mu`	A list of estimated mean vectors.
`probs`	A matrix of posterior probabilities for each observation (rows) and group (columns).
`W`	A binary matrix indicating which entries are considered non-outlying (1 = clean, 0 = outlying). Same dimensions as ‘X‘.
`set_to_zero`	A boolean indicating whether residuals of non-outlying cells should be set to zero.

Details

Positive values of residuals mean that the observed value of the outlying variable is higher than would have been expected based on the other observed variables, negative values mean that the observed value is lower than expected. For non-outlying cells (i.e. where ‘W[i, j] == 1‘), the residual is set to zero.

Value

A numeric matrix of residuals of the same dimension as ‘X‘, where each cell represents the standardized deviation from the model-based conditional expectation, or zero if the cell was not flagged as outlying in ‘W‘.

References

Puchhammer, P., Wilms, I., & Filzmoser, P. (2025). A smooth multi-group Gaussian Mixture Model for cellwise robust covariance estimation. *ArXiv preprint* doi:10.48550/arXiv.2504.02547.

See Also

`cellMGGMM`

Examples

```
data("weatherAUT2021")
cut_lon = c(min(weatherAUT2021$lon)-0.2, 12, 16, max(weatherAUT2021$lon) + 0.2)
cut_lat = c(min(weatherAUT2021$lat)-0.2, 48, max(weatherAUT2021$lat) + 0.2)
groups = ssMRCD::groups_gridbased(weatherAUT2021$lon, weatherAUT2021$lat, cut_lon, cut_lat)
N = length(unique(groups))
model = cellMGGMM(X = weatherAUT2021[, c("p", "s", "vv", "t", "rsum", "rel")],
                  groups = groups,
                  alpha = 0.5)
res = residuals_mggmm(X =  weatherAUT2021[, c("p", "s", "vv", "t", "rsum", "rel")],
                groups = groups,
                Sigma = model$Sigma,
                mu = model$mu,
                probs = model$probs,
                W = model$W)
```

scale.ssMRCD *Locally Center and/or Scale or Data Using an ssMRCD Object*

Description

Applies local standardization (scaling and/or centering) of either the original data from an ssMRCD object or new data provided via the X argument, using group-wise robust means and variances from the ssMRCD estimation.

Usage

```
## S3 method for class 'ssMRCD'
scale(x, ...)
```

Arguments

x An object of class "ssMRCD". See ssMRCD.

... List of additional arguments including:

- X A numeric matrix or data frame containing new observations to be scaled. If not provided, the data stored in the ssMRCD object is used.
- groups An integer vector from 1 to number of groups of group assignments corresponding to the rows of X. If X is not provided, defaults to the group assignments used in the original ssMRCD estimation.
- center_only Logical. If TRUE, only centering is applied; if FALSE, both centering and scaling are applied. Default is FALSE.

Details

For each group, the function applies scaling (or just centering) using the robust location and scale (square root of the diagonal of the covariance) estimates obtained during ssMRCD estimation.

Value

A numeric matrix of the same dimension as X, where each observation has been standardized (or centered) using the corresponding group-wise robust mean and (if applicable) variance from the ssMRCD model. If X = NULL, the original data from the ssMRCD object is returned in scaled form, sorted according to group labels.

See Also

ssMRCD

Examples

```
# Simulated example
x1 <- matrix(runif(200), ncol = 2)
x2 <- matrix(rnorm(200), ncol = 2)
x <- list(x1, x2)

W <- matrix(c(0, 1, 1, 0), ncol = 2)
localCovs <- ssMRCD(x, weights = W, lambda = 0.5)

# Scale original data
sc = scale(localCovs)

# Scale new observations
sc = scale(localCovs,
           list(X = matrix(rnorm(20), ncol = 2, nrow = 10),
           groups = rep(2, 10)))

# Center only
sc = scale(localCovs,
           list(X = matrix(rnorm(20), ncol = 2, nrow = 10),
           groups = rep(2, 10),
           center_only = TRUE))
```

`scores` *Calculate Scores and Distances for Multi-Source PCA*

Description

Computes principal component scores, score distances (SD), and orthogonal distances (OD) for observations grouped into multiple sources using the multi-source PCA model. The function supports either an ‘ssMRCD‘ object for robust local centering and scaling or manually provided group-wise means (‘mu‘) and covariances (‘Sigma‘).

Usage

```
scores(PC, ssMRCD = NULL, X = NULL, groups = NULL, mu = NULL, Sigma = NULL)
```

Arguments

`PC` A 3D array representing the principal component loading matrices for each group (dimensions: variables × components × groups).

`ssMRCD` An optional ‘ssMRCD‘ object used to robustly center and scale ‘X‘. If ‘NULL‘, then ‘X‘, ‘groups‘, ‘mu‘ and ‘Sigma‘, must be provided.

`X` An optional matrix of observations (rows are samples, columns are variables), required if ‘ssMRCD‘ is not provided.

`groups` An optional numeric vector specifying group/source membership for each observation in ‘X‘, required if ‘ssMRCD‘ is not provided.

`mu` Optional list of group-wise means, required if ‘ssMRCD‘ is not provided.

`Sigma` Optional list of group-wise covariance matrices, required if ‘ssMRCD‘ is not provided.

Value

A list with the following components:

scores Matrix of principal component scores for each observation.

SD Numeric vector of score distances, i.e., Mahalanobis distances in the PCA space.

OD Numeric vector of orthogonal distances (reconstruction error orthogonal to PCA space).

X_centered Locally centered input data.

See Also

`ssMRCD`, `scale.ssMRCD`, `msPCA`

Examples

```
# create data set
x1 = matrix(runif(200), ncol = 2)
x2 = matrix(rnorm(200), ncol = 2)
x = list(x1, x2)

# create weighting matrix
W = matrix(c(0, 1, 1, 0), ncol = 2)

# calculate ssMRCD
loccovs = ssMRCD(x, weights = W, lambda = 0.5)

# calculate PCA
pca = msPCA(eta = 1, gamma = 0.5,COVS = loccovs$MRCDcov)

# calculate scores
scores_all = scores(PC = pca$PC, ssMRCD = loccovs)
str(scores_all)

scores_all = scores(PC = pca$PC,
                    X = rbind(x1, x2),
                    groups = rep(c(1,2), each = 100),
                    mu = loccovs$MRCDmu,
                    Sigma = loccovs$MRCDcov)
str(scores_all)
```

screeplot.msPCA *Scree Plot and Cumulative Explained Variance for msPCA Objects*

Description

Generates a scree plot displaying the explained variance of principal components and a heatmap of cumulative explained variance per group for an object of class `msPCA`.

Usage

```
## S3 method for class 'msPCA'
screeplot(x, ...)
```

Arguments

`x` An object of class `msPCA`.

`...` Additional arguments to customize the plot:

`text` Logical; whether to display numeric values on the heatmap (default: `TRUE`).

`size` Numeric; text size for labels in the heatmap (default: 5).

`cutoff` Numeric; cutoff threshold for explained variance lines and color midpoint, as a proportion (default: 0.8).

`gnames` Character vector; optional custom group names (default: `N1`, `N2`, ..., `Nn`).

`textrotate` Numeric; rotation angle of text labels on the heatmap (default: 90 degrees).

Value

A list containing two `ggplot2` plot objects:

- A boxplot-based scree plot showing explained variance per principal component across groups.
- A tile plot showing cumulative explained variance per group and principal component.

Examples

```
set.seed(236)
data <- matrix(rnorm(1500), ncol = 5)
groups <- sample(1:5, 300, replace = TRUE)
W <- time_weights(N = 5, c(3,2,1))
covs <- ssMRCD(data, groups = groups, weights = W, lambda = 0.3)
pca <- msPCA(eta = 0.3, gamma = 0.7, COVS = covs$MRCDcov)
screeplot(pca, text = TRUE, cutoff = 0.8, size = 4, textrotate = 0)
screeplot(pca, text = FALSE, cutoff = 0.6)
```

ssMRCD *Spatially Smoothed MRCD Estimator*

Description

The ssMRCD function calculates the spatially smoothed MRCD estimator from Puchhammer and Filzmoser (2023).

Usage

```
ssMRCD(
  X,
  groups = NULL,
  weights,
  lambda = 0.5,
  tuning = list(method = NULL, plot = FALSE, k = 10, repetitions = 5, cont = 0.05),
  TM = NULL,
  alpha = 0.75,
  maxcond = 50,
  maxcsteps = 200,
  n_initialhsets = NULL
)
```

Arguments

`X` a list of matrices containing the observations per neighborhood sorted, or matrix or data frame containing data. If matrix or data.frame, group vector has to be given.

`groups` vector of neighborhood assignments

`weights` weighting matrix, symmetrical, rows sum up to one and diagonals need to be zero (see also `geo_weights` or `time_weights` .

`lambda` numeric between 0 and 1.

`tuning` default NULL. List of tuning specifications if lambda contains more than one value. See Details.

`TM` target matrix (optional), default value is the covMcd from robustbase.

`alpha` numeric, proportion of values included, between 0.5 and 1.

`maxcond` optional, maximal condition number used for rho-estimation.

`maxcsteps` maximal number of c-steps before algorithm stops.

`n_initialhsets` number of initial h-sets, default is 6 times number of neighborhoods.

Details

The necessary list elements for the parameter `tuning` depend on the method specified. For both tuning approaches (residual-based or contamination-based) the element `method` needs to be specified to `"residuals"` and `"local contamination"`, respectively. The boolean list element `plot` is available for both methods and specifies if a plot should be constructed after tuning.

For `tuning$method = "local contamination"`, additional information needs to be passed. The number of nearest neighbors `tuning$k` used for the local outlier detection method is `10` by default. The percentage of exchanged/contaminated observations is specified by `tuning$cont` and is set to `0.05` by default. Also the coordinates must be given in `tuning$coords` and the number of repetitions for the switching procedure, `tuning$repetitions`.

For `tuning$method = "local contamination"` no optimal value is returned but the choice has to be made by the user. Be aware that the FNR does not take into account that there are also natural outliers included in the data set that might or might not be found. The best parameter selection depends on the goal of the analysis and whether false negatives should be avoided or whether the number of flagged outliers should be low.

Value

The output depends on whether parameters are tuned. If there is no tuning the output is an object of class `"ssMRCD"` containing the following elements:

`MRCDcov` List of ssMRCD-covariance matrices sorted by neighborhood.

`MRCDicov` List of inverse ssMRCD-covariance matrices sorted by neighborhood.

`MRCDmu` List of ssMRCD-mean vectors sorted by neighborhood.

`mX` List of data matrices sorted by neighborhood.

`N` Number of neighborhoods.

`mT` Target matrix.

`rho` Vector of regularization values sorted by neighborhood.

`alpha` Scalar what percentage of observations should be used.

`h` Vector of how many observations are used per neighborhood, sorted.

`numiter` The number of iterations for the best initial h-set combination.

`c_alpha` Consistency factor for normality.

`weights` The weighting matrix.

`lambda` Smoothing factor.

`obj_fun_values` A matrix with objective function values for all initial h-set combinations (rows) and iterations (columns).

`best6pack` initial h-set combinations with best objective function value after c-step iterations.

`Kcov` returns MRCD-estimates without smoothing.

If parameters are tuned, the output consists of:

`ssMRCD` Object of class ssMRCD with optimally selected parameter `lambda`.

`tuning_grid` Vector of lambda to tune over given by the input.

`tuning_values` If `tuning$method` = `"residuals"` then a vector returning the values of the residual criteria for the correspo
If `tuning$method` = `"local contamination"`, then matrix with false negative rates and the total number o

`plot` If `tuning$plot` = `TRUE`, then a plot for parameter tuning is added.

References

Puchhammer P. and Filzmoser P. (2023). Spatially Smoothed Robust Covariance Estimation for Local Outlier Detection. *Journal of Computational and Graphical Statistics, 33(3), 928–940.* doi:10.1080/10618600.2023.2277875

See Also

`plot.ssMRCD`

Examples

```
# create data set
x1 = matrix(runif(200), ncol = 2)
x2 = matrix(rnorm(200), ncol = 2)
x = list(x1, x2)

# create weighting matrix
W = matrix(c(0, 1, 1, 0), ncol = 2)

# calculate ssMRCD
out = ssMRCD(X = x, weights = W, lambda = 0.5)
str(out)
```

summary.msPCA *Summary Method for msPCA Objects*

Description

Provides a summary of a sparse multi-source PCA (‘msPCA‘) result, including group-wise sparsity and explained variance.

Usage

```
## S3 method for class 'msPCA'
summary(object, ...)
```

Arguments

`object`	An object of class `"msPCA"` as returned by the main msPCA fitting function.
`...`	Currently ignored.

Value

Prints a summary to the console. No return value.

`time_weights` *Band weight matrix for time series groupings*

Description

Band weight matrix for time series groupings

Usage

```
time_weights(N, off_diag)
```

Arguments

`N`	number of groups.
`off_diag`	vector for off-diagonal values unequal to zero.

Value

Returns weight matrix for time series groups appropriate for `ssMRCD`.

See Also

`geo_weights`

Examples

```
time_weights(N = 10, off_diag = c(2,1))
```

`weatherAUT2021` *Austrian Weather Data 2021*

Description

This data is a subset of the GeoSphere Austria monthly weather data of 2021 averaged using the median. Stations with missing values are removed.

Usage

```
weatherAUT2021
```

Format

A data frame with 183 rows and 10 columns:

name Unique name of the weather station in German.

lon, lat Longitude and latitude of the weather station.

alt Altitude of the weather station (meter).

p Average air pressure (hPa).

s Monthly sum of sunshine duration (hours).

vv Wind velocity (meter/second).

t Air temperature in 2 meters above the ground in (°C).

rsum Average daily sum of precipitation (mm).

rel Relative air humidity (percent).

Source

The original data was downloaded here (December 2022): `https://data.hub.geosphere.at/dataset/klima-v1-1m`.

References

Data Source: GeoSphere Austria - `https://data.hub.geosphere.at`.

Examples

```
data(weatherAUT2021)
summary(weatherAUT2021)
```

`weatherHoheWarte` *Vienna Weather Time Series (1960-2023)*

Description

This data is a subset of the GeoSphere Austria daily weather data of the time 1960-2023 for the weather station Hohe Warte in Vienna.

Usage

```
weatherHoheWarte
```

Format

A data frame with 23372 rows and 18 columns including 13 weather measurements:

time Time of measurement in date format.

cloud_cover Daily mean of cloud coverage, values between 1 and 100.

global_radiation Daily sum of global radiation (J/cm^2).

vapor_pressure Daily mean of vapour pressuer (hPa).

max_wind_speed Maximal wind speed (m/s).

air_pressure Daily mean of air pressure (hPa).

relative_humidity Daily mean of relative humidity (percent).

precipitation Daily sum of precepitation (mm).

sight Sight distance at 1pm (m).

sunshine_duration Daily sum of sunshine duration (h).

temperature_max Daily maximum of temperature at 2m air height (°C).

temperature_min Daily minimum of temperature at 2m air height (°C).

temperature_mean Daily mean of temperature at 2m air height (°C).

wind_velocity Daily mean of wind speed (m/s).

year Year of measurement.

month Month of measurement.

day Day of the year of measurement.

season Season of measuremen (1 = winter, 2 = spring, 3 = summer, 4 = fall).

Source

The original data was downloaded here (April 2024): `https://data.hub.geosphere.at/dataset/klima-v2-1d`.

References

Data Source: GeoSphere Austria - `https://data.hub.geosphere.at`.

Examples

```
data(weatherHoheWarte)
summary(weatherHoheWarte)
```

Index