Package ‘RaJIVE’

October 12, 2022

Title Robust Angle Based Joint and Individual Variation Explained

Version 1.0

Description A robust alternative to the aJIVE (angle based Joint and Individual Variation Explained) method (Feng et al 2018: <doi:10.1016/j.jmva.2018.03.008>) for the estimation of joint and individual components in the presence of outliers in multi-source data. It decomposes the multi-source data into joint, individual and residual (noise) contributions. The decomposition is robust to outliers and noise in the data. The method is illustrated in Ponzi et al (2021) <arXiv:2101.09110>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.1.0)

Suggests knitr, rmarkdown, testthat (>= 2.1.0), cowplot, reshape2, dplyr

Imports ggplot2, doParallel, foreach

RoxygenNote 7.1.1

NeedsCompilation no

Author Erica Ponzi [aut, cre], Abhik Ghosh [aut]

Maintainer Erica Ponzi <erica.ponzi@medisin.uio.no>

Repository CRAN

Date/Publication 2021-02-04 15:20:05 UTC

R topics documented:

ajive.data.sim 2
data_heatmap 3
decomposition_heatmaps_robustH 4
get_block_loadings 4
get_block_scores 5
get_final_decomposition_robustH 6
get_individual_decomposition_robustH 6

get_individual_rank 7
get_joint_decomposition_robustH 8
get_joint_rank 8
get_joint_scores_robustH 9
get_random_direction_bound_robustH 10
get_svd_robustH 10
get_sv_threshold 11
get_wedin_bound_samples 11
Rajive 12
RobRSVD.all 13
showVarExplained_robust 13
sim_dist 14
svd_reconstruction 14
truncate_svd 15
wedin_bound_resampling 15

Index **16**

`ajive.data.sim` *Simulation of data blocks*

Description

Simulates blocks of data with joint and individual structures

Usage

```
ajive.data.sim(
  K = 3,
  rankJ = 2,
  rankA = c(20, 15, 10),
  n = 100,
  pks,
  dist.type = 1,
  noise = 1
)
```

Arguments

K	Integer. Number of data blocks.
rankJ	Integer. Joint rank.
rankA	Vector of Integers. Individual Ranks.
n	Integer. Number of data points.
pks	Vector of Integers. Number of variables in each block.
dist.type	Integer. 1 for normal, 2 for uniform, 3 for exponential
noise	Integer. Standard deviation in dist

Value

Xsim a list of simulated data matrices and true rank values

Examples

```
n <- 20
p1 <- 10
p2 <- 8
p3 <- 5
JrankTrue <- 2
initial_signal_ranks <- c(5, 2, 2)
 Y <- ajive.data.sim(K =3, rankJ = JrankTrue,
 rankA = initial_signal_ranks,n = n,
 pks = c(p1, p2, p3), dist.type = 1)
```

---

data_heatmap *Decomposition Heatmaps*

Description

Visualization of the RaJIVE decomposition, it shows heatmaps of the decomposition obtained by RaJIVE

Usage

```
data_heatmap(data, show_color_bar = TRUE, title = "", xlab = "", ylab = "")
```

Arguments

data	List. The initial data blocks.
show_color_bar	Boolean.
title	Character.
xlab	Character.
ylab	Character

`decomposition_heatmaps_robustH` *Decomposition Heatmaps*

Description

Visualization of the RaJIVE decomposition, it shows heatmaps of the decomposition obtained by RaJIVE

Usage

```
decomposition_heatmaps_robustH(blocks, jive_results_robust)
```

Arguments

`blocks`	List. The initial data blocks.
`jive_results_robust`	List. The RaJIVE decomposition.

Value

The heatmap of the decomposition

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                  pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
decomposition_heatmaps_robustH(data.ajive, ajive.results.robust)
```

`get_block_loadings` *Block Loadings*

Description

Gets the block loadings from the Rajive decomposition

Usage

```
get_block_loadings(ajive_output, k, type)
```

Arguments

ajive_output	List. The decomposition from Rajive
k	Integer. The index of the data block
type	Character. Joint or individual

Value

The block loadings

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                  pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
get_block_loadings(ajive.results.robust, 2, 'joint')
```

---

get_block_scores *Block Scores*

Description

Gets the block scores from the Rajive decomposition

Usage

```
get_block_scores(ajive_output, k, type)
```

Arguments

ajive_output	List. The decomposition from Rajive
k	Integer. The index of the data block
type	Character. Joint or individual

Value

The block scores

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                 pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
get_block_scores(ajive.results.robust, 2, 'joint')
```

`get_final_decomposition_robustH`
Computes the final JIVE decomposition.

Description

Computes X = J + I + E for a single data block and the respective SVDs.

Usage

```
get_final_decomposition_robustH(X, joint_scores, sv_threshold, full = TRUE)
```

Arguments

`X`	Matrix. The original data matrix.
`joint_scores`	Matrix. The basis of the joint space (dimension n x joint_rank).
`sv_threshold`	Numeric vector. The singular value thresholds from the initial signal rank estimates.
`full`	Boolean. Do we compute the full J, I matrices or just svd

---

`get_individual_decomposition_robustH`
Computes the individual matrix for a data block.

Description

Computes the individual matrix for a data block.

Usage

```
get_individual_decomposition_robustH(
  X,
  joint_scores,
  sv_threshold,
  full = TRUE
)
```

Arguments

`X`	Matrix. The original data matrix.
`joint_scores`	Matrix. The basis of the joint space (dimension n x joint_rank).
`sv_threshold`	Numeric vector. The singular value thresholds from the initial signal rank estimates.
`full`	Boolean. Do we compute the full J, I matrices or just the SVD (set to FALSE to save memory).

`get_individual_rank` *Individual Rank*

Description

Gets the individual ranks from the Rajive decomposition

Usage

```
get_individual_rank(ajive_output, k)
```

Arguments

`ajive_output`	List. The decomposition from Rajive
`k`	Integer. The index of the data block.

Value

The individual ranks

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                 pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
get_individual_rank(ajive.results.robust, 2)
```

get_joint_decomposition_robustH *Computes the individual matrix for a data block*

Description

Computes the individual matrix for a data block

Usage

```
get_joint_decomposition_robustH(X, joint_scores, full = TRUE)
```

Arguments

`X`	Matrix. The original data matrix.
`joint_scores`	Matrix. The basis of the joint space (dimension n x joint_rank).
`full`	Boolean. Do we compute the full J, I matrices or just the SVD (set to FALSE to save memory).

get_joint_rank *Joint Rank*

Description

Gets the joint rank from the Rajive decomposition

Usage

```
get_joint_rank(ajive_output)
```

Arguments

`ajive_output`	List. The decomposition from Rajive

Value

The joint rank

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                   pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
get_joint_rank(ajive.results.robust)
```

---

get_joint_scores_robustH

Computes the joint scores.

Description

Estimate the joint rank with the wedin bound, compute the signal scores SVD, double check each joint component.

Usage

```
get_joint_scores_robustH(
  blocks,
  block_svd,
  initial_signal_ranks,
  sv_thresholds,
  n_wedin_samples = 1000,
  n_rand_dir_samples = 1000,
  joint_rank = NA
)
```

Arguments

`blocks`	List. A list of the data matrices.
`block_svd`	List. The SVD of the data blocks.
`initial_signal_ranks`	Numeric vector. Initial signal ranks estimates.
`sv_thresholds`	Numeric vector. The singular value thresholds from the initial signal rank estimates.
`n_wedin_samples`	Integer. Number of wedin bound samples to draw for each data matrix.
`n_rand_dir_samples`	Integer. Number of random direction bound samples to draw.
`joint_rank`	Integer or NA. User specified joint_rank. If NA will be estimated from data.

get_random_direction_bound_robustH

Estimate the wedin bound for a data matrix.

Description

Samples from the random direction bound. Returns on the scale of squared singular value.

Usage

```
get_random_direction_bound_robustH(n_obs, dims, num_samples = 1000)
```

Arguments

`n_obs`	The number of observations.
`dims`	The number of features in each data matrix
`num_samples`	Integer. Number of vectors selected for resampling procedure.

Value

rand_dir_samples

get_svd_robustH

Computes the robust SVD of a matrix Using robRsvd

Description

Computes the robust SVD of a matrix Using robRsvd

Usage

```
get_svd_robustH(X, rank = NULL)
```

Arguments

`X`	Matrix. X matrix.
`rank`	Integer. Rank of SVD decomposition

Value

List. The SVD of X.

`get_sv_threshold` *The singular value threshold.*

Description

Computes the singular value threshold for the data matrix (half way between the rank and rank + 1 singluar value).

Usage

```
get_sv_threshold(singular_values, rank)
```

Arguments

`singular_values`	Numeric. The singular values.
`rank`	Integer. The rank of the approximation.

`get_wedin_bound_samples` *Gets the wedin bounds*

Description

Gets the wedin bounds

Usage

```
get_wedin_bound_samples(X, SVD, signal_rank, num_samples = 1000)
```

Arguments

`X`	Matrix. The data matrix.
`SVD`	List. The SVD decomposition of the matrix. List with entries 'u', 'd', and 'v'from the svd function.
`signal_rank`	Integer.
`num_samples`	Integer. Number of vectors selected for resampling procedure.

Rajive *Robust Angle based Joint and Individual Variation Explained*

Description

Computes the robust aJIVE decomposition with parallel computation.

Usage

```
Rajive(
  blocks,
  initial_signal_ranks,
  full = TRUE,
  n_wedin_samples = 1000,
  n_rand_dir_samples = 1000,
  joint_rank = NA
)
```

Arguments

`blocks`	List. A list of the data matrices.
`initial_signal_ranks`	Vector. The initial signal rank estimates.
`full`	Boolean. Whether or not to store the full J, I, E matrices or just their SVDs (set to FALSE to save memory).
`n_wedin_samples`	Integer. Number of wedin bound samples to draw for each data matrix.
`n_rand_dir_samples`	Integer. Number of random direction bound samples to draw.
`joint_rank`	Integer or NA. User specified joint_rank. If NA will be estimated from data.

Value

The aJIVE decomposition.

Examples

```
n <- 50
pks <- c(100, 80, 50)
Y <- ajive.data.sim(K =3, rankJ = 3, rankA = c(7, 6, 4), n = n,
                    pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 6, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]), (Y$sim_data[[3]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
```

RobRSVD.all — *Computes the robust SVD of a matrix*

Description

Computes the robust SVD of a matrix

Usage

```
RobRSVD.all(data, nrank = min(dim(data)), svdinit = svd(data))
```

Arguments

data	Matrix. X matrix.
nrank	Integer. Rank of SVD decomposition
svdinit	List. The standard SVD.

Value

List. The SVD of X.

showVarExplained_robust — *Proportions of variance explained*

Description

Gets the variance explained by each component of the Rajive decomposition

Usage

```
showVarExplained_robust(ajiveResults, blocks)
```

Arguments

ajiveResults	List. The decomposition from Rajive
blocks	List. The initial data blocks

Value

The proportion of variance explained by each component

Examples

```
n <- 10
pks <- c(20, 10)
Y <- ajive.data.sim(K =2, rankJ = 2, rankA = c(7, 4), n = n,
                  pks = pks, dist.type = 1)
initial_signal_ranks <-  c(7, 4)
data.ajive <- list((Y$sim_data[[1]]), (Y$sim_data[[2]]))
ajive.results.robust <- Rajive(data.ajive, initial_signal_ranks)
showVarExplained_robust(ajive.results.robust, data.ajive)
```

`sim_dist` *Simulation of single data block from distribution*

Description

Simulation of single data block from distribution

Usage

```
sim_dist(num, n, p)
```

Arguments

`num`	Integer. Type of distribution. 1 for normal, 2 for uniform, 3 for exponential
`n`	Integer. Number of data points.
`p`	Integers. Number of variables in block.

`svd_reconstruction` *Reconstruces the original matrix from its robust SVD.*

Description

Computes UDV^T to get the approximate (or full) X matrix.

Usage

```
svd_reconstruction(decomposition)
```

Arguments

`decomposition`	List. List with entries 'u', 'd', and 'v'from the svd function.

Value

Matrix. The original matrix.

truncate_svd *Truncates a robust SVD.*

Description

Removes columns from the U, D, V matrix computed form an SVD.

Usage

```
truncate_svd(decomposition, rank)
```

Arguments

`decomposition`	List. List with entries 'u', 'd', and 'v'from the svd function.
`rank`	List. List with entries 'u', 'd', and 'v'from the svd function.

Value

The trucated robust SVD of X.

wedin_bound_resampling *Resampling procedure for the wedin bound*

Description

Resampling procedure for the wedin bound

Usage

```
wedin_bound_resampling(X, perp_basis, right_vectors, num_samples = 1000)
```

Arguments

`X`	Matrix. The data matrix.
`perp_basis`	Matrix. Either U_perp or V_perp: the remaining left/right singluar vectors of X after estimating the signal rank.
`right_vectors`	Boolean. Right multiplication or left multiplication.
`num_samples`	Integer. Number of vectors selected for resampling procedure.

Index

ajive.data.sim, 2

data_heatmap, 3
decomposition_heatmaps_robustH, 4

get_block_loadings, 4
get_block_scores, 5
get_final_decomposition_robustH, 6
get_individual_decomposition_robustH, 6
get_individual_rank, 7
get_joint_decomposition_robustH, 8
get_joint_rank, 8
get_joint_scores_robustH, 9
get_random_direction_bound_robustH, 10
get_sv_threshold, 11
get_svd_robustH, 10
get_wedin_bound_samples, 11

Rajive, 12
RobRSVD.all, 13

showVarExplained_robust, 13
sim_dist, 14
svd_reconstruction, 14

truncate_svd, 15

wedin_bound_resampling, 15