

Package ‘rSDR’

November 10, 2025

Title Robust Sufficient Dimension Reduction

Version 1.0.3.0

Description

A novel sufficient-dimension reduction method is robust against outliers using alpha-distance covariance and manifold-learning in dimensionality reduction problems. Please refer Hsin-Hsiung Huang, Feng Yu & Teng Zhang (2024) <[doi:10.1080/10485252.2024.2313137](https://doi.org/10.1080/10485252.2024.2313137)> for the details.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Imports ManifoldOptim, methods, Rcpp, rstiefel, scatterplot3d, future, future.apply, ggplot2, ggsci

Suggests expm, knitr, rmarkdown, Matrix, RcppNumerical, fdm2id

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Sheau-Chiann Chen [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-5574-1434>>),
Shilin Zhao [aut],
Hsin-Hsiung Bill Huang [aut] (ORCID:
<<https://orcid.org/0000-0001-7150-7229>>)

Maintainer Sheau-Chiann Chen <sheau-chiann.chen.1@vumc.org>

Repository CRAN

Date/Publication 2025-11-10 00:10:02 UTC

Contents

optimal_alpha_boot	2
optimal_alpha_cv	3
plot_alpha	4
plot_rSDR	5
rSDR	6

optimal_alpha_boot	<i>The optimal alpha for rSDR via bootstrap resampling</i>
--------------------	--

Description

Perform R bootstrap replications of the cost function in rSDR method and return the corresponding optimal alpha.

Usage

```
optimal_alpha_boot(alpha.v,X,Y,d,R,maxiter=1000,tol=1e-7)
```

Arguments

alpha.v	user-supplied alpha sequence. The default is <code>alpha.v=c(0.3,0.4,0.5,0.6,0.7)</code> .
X	an $n \times p$ numeric matrix, where n is the number of observations and p is the number of variable.
Y	an $n \times k$ numeric response matrix, where $k(\geq 1)$ is the number of response variables.
d	the number of reduced dimension. The default is <code>d=3</code> .
R	the number of bootstrap replicates.
maxiter	maxiter is the maximum number of iterations allowed for the solver (a non-negative integer). See the <code>Max_Iteration</code> parameter in get.solver.params for details.
tol	tol is used to assess convergence, see the <code>Tolerance</code> parameter in get.solver.params for details.

Value

An object of class "optimal_alpha_boot" is returned. The returned value contains the following components:

opt.alpha value of alpha that gives minimum `f_test.meam`.

f_test.mean The mean of cost function by the alpha sequence - a vector of length `length(alpha.v)`.

f_test.sd The standard deviation of cost function by the alpha sequence.

f_test An $R \times \text{length}(\text{alpha.v})$ matrix. The cost value for each fold at a given alpha.

d The value of d as passed to `optimal_alpha_boot`.

R The value of R as passed to `optimal_alpha_boot`.

Examples

```

library(ManifoldOptim)
library(rSDR)
library(future)
library(future.apply)
utils::data("ionosphere", package = "fddm2id")
X<-as.matrix(ionosphere[,c(1:33)])
Y<-ifelse(ionosphere[,34]=='b',0,1)
Y<-matrix(Y,length(Y),1)
set.seed(2435)
# plan(multisession) will launch parallel workers running in the background
# to save running time. To shut down background workers launched this way, call
# plan(sequential)
# use all local cores except one
# future::plan(future::multisession, workers = future::availableCores() - 1)
# use 2 cores for parallel

future::plan("multisession", workers = 2)
opt_results<-optimal_alpha_boot(alpha.v=c(0.3,0.5,0.7),X=X,Y=Y,d=3,R=5)
opt_results
future::plan(sequential)

```

optimal_alpha_cv	<i>The optimal alpha for rSDR via cross-validation</i>
------------------	--

Description

Performs k-folds cross-validation for rSDR method and returns the corresponding optimal alpha.

Usage

```
optimal_alpha_cv(alpha.v,X,Y,d,kfolds=10,maxiter=1000,tol=1e-7)
```

Arguments

alpha.v	user-supplied alpha sequence. The default is <code>alpha.v=c(0.3,0.4,0.5,0.6,0.7)</code> .
X	an $n \times p$ numeric matrix, where n is the number of observations and p is the number of variable.
Y	an $n \times k$ numeric response matrix, where $k(\geq 1)$ is the number of response variables.
d	the number of reduced dimension. The default is <code>d=3</code> .
kfolds	the number of folds - default is 10.
maxiter	maxiter is the maximum number of iterations allowed for the solver (a non-negative integer). See the <code>Max_Iteration</code> parameter in get.solver.params for details.
tol	tol is used to assess convergence, see the <code>Tolerance</code> parameter in get.solver.params for details.

Value

An object of class "optimal_alpha_cv" is returned. The returned value contains the following components:

opt.alpha value of alpha that gives minimum `f_test.meam`.

f_test.mean The mean of cost value by the alpha sequence - a vector of length `length(alpha.v)`.

f_test.sd The standard deviation of cost value by the alpha sequence - a vector of length `length(alpha.v)`.

f_test A `kfolds × length(alpha.v)` matrix. The cost value for each fold at a given alpha.

d The value of `d` as passed to `optimal_alpha_cv`.

kfolds The value of `kfolds` as passed to `optimal_alpha_cv`.

Examples

```
library(ManifoldOptim)
library(rSDR)
library(future)
library(future.apply)
utils::data("ionosphere", package = "fddm2id")
X<-as.matrix(ionosphere[,c(1:33)])
Y<-ifelse(ionosphere[,34]=='b',0,1)
Y<-matrix(Y,length(Y),1)
set.seed(2435)
# plan(multisession) will launch parallel workers running in the background
# to save running time. To shut down background workers launched this way, call
# plan(sequential)
# use all local cores except one
# future::plan(future::multisession, workers = future::availableCores() - 1)
# use 2 cores for parallel

future::plan("multisession", workers = 2)
opt_results<-optimal_alpha_cv(alpha.v=c(0.3, 0.5, 0.7),X=X,Y=Y,d=3,kfolds=5)
opt_results
```

plot_alpha

Plot for the optimal alpha

Description

Plot for the mean with the standard deviation of cost function and alpha

Usage

```
plot_alpha(opt_results)
```

Arguments

`opt_results` `opt_results` is from either `optimal_alpha_boot` or `optimal_alpha_cv`

Value

No return value, showing the mean and standard deviation of cost function for each alpha value.

Examples

```
library(ManifoldOptim)
library(rSDR)
utils::data("ionosphere", package = "fdm2id")
X<-as.matrix(ionosphere[,c(1:33)])
Y<-ifelse(ionosphere[,34]=='b',0,1)
Y<-matrix(Y,length(Y),1)
set.seed(2435)
# plan(multisession) will launch parallel workers running in the background
# to save running time. To shut down background workers launched this way, call
# plan(sequential)
# use all local cores except one
# future::plan(future::multisession, workers = future::availableCores() - 1)
# use 2 cores for parallel

future::plan("multisession", workers = 2)
opt_results<-optimal_alpha_cv(alpha.v=c(0.3, 0.5, 0.7),X=X,Y=Y,d=3,kfolds=10)
plot_alpha(opt_results=opt_results)
```

plot_rSDR

Projected data plotting

Description

Function for plotting of projected_data from rSDR results.

Usage

```
plot_rSDR(projected_data,Y,Y.name,colors=NULL)
```

Arguments

projected_data	projected data from rSDR results.
Y	an $n \times 1$ numeric matrix.
Y.name	label for y-axis
colors	Assign specific colors to each level of the response variable.

Value

No return value, visualizing reduced-dimensional data using 1D, 2D, or 3D projections. When the reduced dimension exceeds three, pairwise scatter plots are automatically generated.

Examples

```
library(ManifoldOptim)
library(rSDR)
utils::data("ionosphere", package = "fddm2id")
X<-as.matrix(ionosphere[,c(1:33)])
Y<-ifelse(ionosphere[,34]=='b',0,1)
Y<-matrix(Y,length(Y),1)
ionosphere$V35<-factor(ionosphere$V35,levels=c('b','g'),labels=c('Bad','Good'))
set.seed(2435)

sdr_result<-rSDR(X=X, Y=Y, d=3, alpha=0.3,maxiter=1000,tol=1e-7)
plot_rSDR(projected_data=sdr_result$projected_data,Y=ionosphere$V35,
Y.name='group',colors=c("#374E55FF", "#DF8F44FF"))
```

rSDR

Robust Sufficient Dimension Reduction

Description

Robust Sufficient Dimension Reduction with alpha-Distance Covariance and Stiefel Manifold Learning for supervised dimension reduction.

Usage

```
rSDR(X, Y, d, alpha=0.5,maxiter=1000,tol=1e-7)
```

Arguments

X	an $n \times p$ numeric matrix, where n is the number of observations and p is the number of variable.
Y	an $n \times k$ numeric response matrix, where $k(\geq 1)$ is the number of response variables.
d	the number of reduced dimension.
alpha	this parameter represents the exponent applied to the Euclidean distance in the computation of distance covariance. When $\alpha=1$, it corresponds to the classical distance covariance. When $0 < \alpha < 1$, it is a more robust version by reducing the influence of large values in the distance matrices.
maxiter	maxiter is the maximum number of iterations allowed for the solver (a non-negative integer). See the Max_Iteration parameter in get.solver.params for details.
tol	tol is used to assess convergence, see the Tolerance parameter in get.solver.params for details.

Value

The returned value is an object of class "rSDR", containing the following components:

projected_data an $n \times d$ matrix representing the projected data using the rSDR method.

beta a $p \times d$ matrix. Solve β by $\mathbf{C} = \Sigma_x^{1/2} \beta$

C_value an optimal of C is obtained by maximizing the target function using ManifoldOptim method.

f_value The value of cost function f is defined as the negative of the target function.

References

Hsin-Hsiung Huang, Feng Yu & Teng Zhang (19 Feb 2024): Robust sufficient dimension reduction via alpha-distance covariance, Journal of Nonparametric Statistics, DOI:10.1080/10485252.2024.2313137

Examples

```
library(ManifoldOptim)
library(rSDR)
utils::data("ionosphere", package = "fddm2id")
X<-as.matrix(ionosphere[,c(1:33)])
Y<-ifelse(ionosphere[,34]=='b',0,1)
Y<-matrix(Y,length(Y),1)
set.seed(2435)

sdr_result<-rSDR(X=X, Y=Y, d=3, alpha=0.3,maxiter=1000,tol=1e-7)
```

Index

`get.solver.params`, [2](#), [3](#), [6](#)

`optimal_alpha_boot`, [2](#), [4](#)

`optimal_alpha_cv`, [3](#), [4](#)

`plot_alpha`, [4](#)

`plot_rSDR`, [5](#)

`rSDR`, [6](#)