Package ‘msma’

February 14, 2024

Type Package
Title Multiblock Sparse Multivariable Analysis
Version 3.1
Date 2024-02-13
Author Atsushi Kawaguchi
Maintainer Atsushi Kawaguchi <kawa_a24@yahoo.co.jp>
Depends R (>= 3.5)
Description Several functions can be used to analyze multiblock multivariable data. If the input is a single matrix, then principal components analysis (PCA) is implemented. If the input is a list of matrices, then multiblock PCA is implemented. If the input is two matrices, for exploratory and objective variables, then partial least squares (PLS) analysis is implemented. If the input is two lists of matrices, for exploratory and objective variables, then multiblock PLS analysis is implemented. Additionally, if an extra outcome variable is specified, then a supervised version of the methods above is implemented. For each method, sparse modeling is also incorporated. Functions for selecting the number of components and regularized parameters are also provided.
License GPL (>= 2)
Suggests knitr, rmarkdown
VignetteBuilder knitr
NeedsCompilation no
Repository CRAN
Date/Publication 2024-02-14 05:30:13 UTC

R topics documented:

msma-package .................................................. 2
cvmsma .......................................................... 2
hcmsma .......................................................... 4
msma ............................................................ 5
ncompsel ......................................................... 8
optparasearch ................................................. 11
msma-package

Multiblock Sparse Matrix Analysis Package

Description

A Package for Implementation of the method

Author(s)

Atsushi Kawaguchi. <kawa_a24@yahoo.co.jp>

References


See Also

msma

cvmsma

Cross-Validation

Description

cross-validated method to evaluate the fit of "msma".

Usage

cvmsma(
  X,
  Y = NULL,
  Z = NULL,
  comp = 1,
  lambdaX,
  lambdaY = NULL,
  lambdaXsup = NULL,
  lambdaYsup = NULL,
cvmsma

eta = 1,
type = "lasso",
inX = NULL,
inY = NULL,
inXsup = NULL,
inYsup = NULL,
muX = 0,
muY = 0,
nfold = 5,
seed = 1,
intseed = 1
)

Arguments

X
a (list of) matrix, explanatory variable(s).

Y
a (list of) matrix, objective variable(s).

Z
a (list of) matrix, response variable(s).

comp
numeric scalar for the maximum number of components to be considered.

lambdaX
numeric vector of regularized parameters for X with length equal to the number of blocks. If omitted, no regularization is conducted.

lambdaY
numeric vector of regularized parameters for Y with length equal to the number of blocks. If omitted, no regularization is conducted.

lambdaXsup
numeric vector of regularized parameters for the super weight of X with length equal to the number of blocks. If omitted, no regularization is conducted.

lambdaYsup
numeric vector of regularized parameters for the super weight of Y with length equal to the number of blocks. If omitted, no regularization is conducted.

eta
numeric scalar the parameter indexing the penalty family.

type
a character.

inX
a (list of) numeric vector to specify the variables of X which are always in the model.

inY
a (list of) numeric vector to specify the variables of X which are always in the model.

inXsup
a (list of) numeric vector to specify the blocks of X which are always in the model.

inYsup
a (list of) numeric vector to specify the blocks of Y which are always in the model.

muX
a numeric scalar for the weight of X for the supervised.

muY
a numeric scalar for the weight of Y for the supervised.

nfold
number of folds - default is 5.

seed
number of seed for the random number.

intseed
seed number for the random number in the parameter estimation algorithm.
Details

k-fold cross-validation for msma

Value

err       The mean cross-validated errors which has three elements consisting of the mean of errors for X and Y, the errors for X and for Y.

Examples

##### data #####
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y

##### One Component CV #####
cv1 = cvmsma(X, Y, comp = 1, lambdaX=2, lambdaY=1:3, nfold=5, seed=1)
cv1

##### Two Component CV #####
cv2 = cvmsma(X, Y, comp = 2, lambdaX=2, lambdaY=1:3, nfold=5, seed=1)
cv2

hcmsma

Hierarchical cluster analysis

Description

This is a function for performing a hierarchical cluster analysis using scores

Usage

hcmsma(
  object,
  nclust = 4,
  graph = FALSE,
  hmethod = "ward.D2",
  axes = c(1, 2),
  block = "block",
  XY = "X"
)

Arguments

object       an object of class "msma", usually, a result of a call to msma
nclust       a numeric scalar, number of clusters.
graph        a numeric vector, numbers of columns for Y. The length of vector corresponds to the number of blocks.
msma

- **hmethod**: the agglomeration method to be used in the function "hclust".
- **axes**: a numeric (or vector), specifying the component(s) to analyze.
- **block**: a character, indicating which the "block" or "super" is used.
- **XY**: a character, indicating "X" or "Y".

**Details**

This function performs a hierarchical cluster analysis using scores.

**Value**

- **hcout**: An object of class hclust
- **clusters**: a vector with group memberships
- **object**: an object of class "msma", usually, a result of a call to *msma*

---

**msma**

*Multiblock Sparse Partial Least Squares*

**Description**

This is a function for a matrix decomposition method incorporating sparse and supervised modeling for a multiblock multivariable data analysis.

**Usage**

```r
msma(X, ...)
```

### Default S3 method:
```r
msma(
  X,
  Y = NULL,
  Z = NULL,
  comp = 2,
  lambdaX = NULL,
  lambdaY = NULL,
  lambdaXsup = NULL,
  lambdaYsup = NULL,
  eta = 1,
  type = "lasso",
  inX = NULL,
  inY = NULL,
  inXsup = NULL,
  inYsup = NULL,
  muX = 0,
  muY = 0,
  defmethod = "canonical",
)```
msma

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

### Arguments

- **X**: a matrix or list of matrices indicating the explanatory variable(s). This parameter is required.
- **Y**: a matrix or list of matrices indicating objective variable(s). This is optional. If there is no input for Y, then PCA is implemented.
- **Z**: a vector, response variable(s) for implementing the supervised version of (multi-block) PCA or PLS. This is optional. The length of Z is the number of subjects. If there is no input for Z, then unsupervised PLS/PCA is implemented.
- **comp**: numeric scalar for the maximum number of components to be considered.
- **lambdaX**: numeric vector of regularized parameters for X, with a length equal to the number of blocks. If lambdaX is omitted, no regularization is conducted.
- **lambdaY**: numeric vector of regularized parameters for Y, with a length equal to the number of blocks. If lambdaY is omitted, no regularization is conducted.
- **lambdaXsup**: numeric vector of regularized parameters for the super weight of X with length equal to the number of blocks. If omitted, no regularization is conducted.
- **lambdaYsup**: numeric vector of regularized parameters for the super weight of Y with length equal to the number of blocks. If omitted, no regularization is conducted.
- **eta**: numeric scalar indicating the parameter indexing the penalty family. This version contains only choice 1.
- **type**: a character, indicating the penalty family. In this version, only one choice is available: "lasso."
- **inX**: a vector or list of numeric vectors specifying the variables in X, always included in the model
- **inY**: a vector or list of numeric vectors specifying the variables in Y, always included in the model
- **inXsup**: a (list of) numeric vector to specify the blocks of X which are always in the model.
- **inYsup**: a (list of) numeric vector to specify the blocks of Y which are always in the model.
- **muX**: a numeric scalar for the weight of X for the supervised case. $0 \leq \mu_X \leq 1$.
- **muY**: a numeric scalar for the weight of Y for the supervised case. $0 \leq \mu_Y \leq 1$.\n
```r
scaling = TRUE,
verbose = FALSE,
intseed = 1,
ceps = 1e-04,
...}
```

### Notes

- The function `msma` is a method for the S3 class `msma`, providing a flexible framework for analyzing multi-block datasets. It supports various penalty families, including the lasso, and allows for regularization of the data matrices. The function is designed to handle both supervised and unsupervised learning scenarios, with options to specify different components, regularization parameters, and model specifications.

- The `X` parameter is a matrix or list of matrices indicating the explanatory variables, which are essential for the model.

- The `Y` parameter is optional and can be a matrix or list of matrices indicating the objective variables, enabling the implementation of PCA or PLS.

- The `Z` parameter is used for specifying the response variable(s) in the case of supervised learning, where a vector is provided to indicate the number of subjects.

- The `comp` parameter is used to specify the maximum number of components to be considered in the model.

- Regularization parameters (`lambdaX`, `lambdaY`, `lambdaXsup`, `lambdaYsup`) allow for controlling the degree of regularization in the model, which is crucial for preventing overfitting.

- The `eta` parameter is used to index the penalty family, currently limited to the lasso choice.

- Additional arguments (`...`) can be passed to or from other methods, providing flexibility for further customization.

- The `print` method is used to display the structure and components of objects of class `msma`, aiding in the interpretation and visualization of the model results.
defmethod a character representing the deflation method. This version has only the choice "canonical."
scaling a logical, indicating whether or not data scaling is performed. The default is TRUE.
verbose information
intseed seed number for the random number in the parameter estimation algorithm.
ceps a numeric scalar for the convergence condition of the algorithm
x an object of class "msma", usually, a result of a call to msma

Details

msma requires at least one input X (a matrix or list). In this case, (multiblock) PCA is conducted. If Y is also specified, then a PLS is conducted using X as explanatory variables and Y as objective variables. This function scales each data matrix to a mean of 0 and variance of 1 in the default. The block structure can be represented as a list. If Z is also specified, a supervised version is implemented, and the degree is controlled by muX or muY, where 0 <= muX <= 1, 0 <= muY <= 1, and 0 <= muX + muY < 1. If a positive lambdaX or lambdaY is specified, then a sparse estimation based on the L1 penalty is implemented.

Value
dmode Which modes "PLS" or "PCA"
X Scaled X which has a list form.
Y Scaled Y which has a list form.
Xscale Scaling information for X. The means and standard deviations for each block of X are returned.
Yscale Scaling information for Y. The means and standard deviations for each block of Y are returned.
comp the number of componets
wbX block loading for X
sbX block score for X
wbY block loading for Y
sbY block score for Y
ssX super score for X
wsX super loading for X
ssY super score for Y
wsY super loading for Y
nzwbX number of nonzeros in block loading for X
nzwbY number of nonzeros in block loading for Y
nzwsX number of nonzeros in super loading for X
nzwsY number of nonzeros in super loading for Y
selectXnames names of selected variables for X
**selectYnames**  
Names of selected variables for Y.

**avX**  
The adjusted variance of the score for X.

**avY**  
The adjusted variance of the score for Y.

**cpevX**  
The cumulative percentage of the explained variance for X.

**cpevY**  
The cumulative percentage of the explained variance for Y.

**reproduct**  
Predictivity. Correlation between Y and the predicted Y.

**predictiv**  
Reproductivity. Correlation between the score for Y and the outcome Z.

### Examples

#### data ####
```r
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y
```

#### One Component ####
```r
fit1 = msma(X, Y, comp=1, lambdaX=2, lambdaY=1:3)
fit1
```

#### Two Component ####
```r
fit2 = msma(X, Y, comp=2, lambdaX=2, lambdaY=1:3)
fit2
```

#### Sparse Principal Component Analysis ####
```r
fit3 = msma(X, comp=5, lambdaX=2.5)
summary(fit3)
```

### ncompsearch

**Search for Number of Components**

**Description**

Determination of the number of components based on cross-validated method or Bayesian information criterion (BIC).

**Usage**

```r
ncompsearch(
X,
Y = NULL,
Z = NULL,
comps = 1:3,
lambdaX = NULL,
lambdaY = NULL,
lambdaXsup = NULL,
lambdaYsup = NULL,
eta = 1,
)```
ncompsearch

type = "lasso",
inX = NULL,
inY = NULL,
inXsup = NULL,
inYsup = NULL,
mux = 0,
muy = 0,
nfold = 5,
regpara = FALSE,
maxrep = 3,
minpct = 0,
maxpct = 1,
criterion = c("CV", "BIC")[1],
whichselect = NULL,
intseed = 1
)

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

## S3 method for class 'ncompsearch'
plot(x, cidx = 1, ...)

Arguments

X a matrix or list of matrices indicating the explanatory variable(s). This parameter is required.

Y a matrix or list of matrices indicating objective variable(s). This is optional. If there is no input for Y, then PCA is implemented.

Z a vector, response variable(s) for implementing the supervised version of (multi-block) PCA or PLS. This is optional. The length of Z is the number of subjects. If there is no input for Z, then unsupervised PLS/PCA is implemented.

comps numeric vector for the maximum numbers of components to be considered.

lambdaX numeric vector of regularized parameters for X, with a length equal to the number of blocks. If lambdaX is omitted, no regularization is conducted.

lambdaY numeric vector of regularized parameters for Y, with a length equal to the number of blocks. If lambdaY is omitted, no regularization is conducted.

lambdaXsup numeric vector of regularized parameters for the super weight of X with length equal to the number of blocks. If omitted, no regularization is conducted.

lambdaYsup numeric vector of regularized parameters for the super weight of Y with length equal to the number of blocks. If omitted, no regularization is conducted.

eta numeric scalar indicating the parameter indexing the penalty family. This version contains only choice 1.

type a character, indicating the penalty family. In this version, only one choice is available: "lasso."
\texttt{inX}  
a (list of) numeric vector to specify the variables of X which are always in the model.

\texttt{inY}  
a (list of) numeric vector to specify the variables of X which are always in the model.

\texttt{inXsup}  
a (list of) numeric vector to specify the blocks of X which are always in the model.

\texttt{inYsup}  
a (list of) numeric vector to specify the blocks of Y which are always in the model.

\texttt{muX}  
a numeric scalar for the weight of X for the supervised case. $0 \leq \mu_X \leq 1$.

\texttt{muY}  
a numeric scalar for the weight of Y for the supervised case. $0 \leq \mu_Y \leq 1$.

\texttt{nfold}  
number of folds - default is 5.

\texttt{regpara}  
logical, If TRUE, the regularized parameters search is also conducted simultaneously.

\texttt{maxrep}  
numeric scalar for the number of iteration.

\texttt{minpct}  
minimum candidate parameters defined as a percentile of automatically determined (possible) candidates.

\texttt{maxpct}  
maximum candidate parameters defined as a percentile of automatically determined (possible) candidates.

\texttt{criterion}  
a character, the evaluation criterion, "CV" for cross-validation, based on a matrix element-wise error, and "BIC" for Bayesian information criteria. The "BIC" is the default.

\texttt{whichselect}  
which blocks selected.

\texttt{intseed}  
seed number for the random number in the parameter estimation algorithm.

\texttt{x}  
an object of class "ncompsearch", usually, a result of a call to \texttt{ncompsearch}

\texttt{...}  
further arguments passed to or from other methods.

\texttt{cidx}  
Parameters used in the plot function to specify whether block or super is used. 1=block (default), 2=super.

\textbf{Details}

This function searches for the optimal number of components.

\textbf{Value}

\begin{tabular}{ll}
\texttt{comps} & numbers of components \\
\texttt{mincriterion} & minimum criterion values \\
\texttt{ criterions} & criterion values \\
\texttt{optncomp} & optimal number of components based on minimum cross-validation error
\end{tabular}
Examples

##### data #####

tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y

##### number of components search #####
ncomp1 = ncompsearch(X, Y, comps = c(1, 5, 10*(1:2)), nfold=5)
#plot(ncomp1)

---

**optparasearch**  
*Parameters Search*

**Description**

Combined method for optimizing the number of components and regularized parameters for "msma".

**Usage**

```r
optparasearch(
  X,
  Y = NULL,
  Z = NULL,
  search.method = c("ncomp1st", "regpara1st", "regparaonly", "simultaneous")[1],
  eta = 1,
  type = "lasso",
  inX = NULL,
  inY = NULL,
  muX = 0,
  muY = 0,
  comp = 10,
  nfold = 5,
  maxrep = 3,
  minpct = 0,
  maxpct = 1,
  maxpct4ncomp = NULL,
  criterion = c("BIC", "CV")[1],
  criterion4ncomp = NULL,
  whichselect = NULL,
  homo = NULL,
  intseed = 1
)
```

## S3 method for class 'optparasearch'

print(x, ...)
Arguments

X: a matrix or list of matrices indicating the explanatory variable(s). This parameter is required.

Y: a matrix or list of matrices indicating objective variable(s). This is optional. If there is no input for Y, then PCA is implemented.

Z: a vector, response variable(s) for implementing the supervised version of (multiblock) PCA or PLS. This is optional. The length of Z is the number of subjects. If there is no input for Z, then unsupervised PLS/PCA is implemented.

search.method: a character indicating search methods, see Details. Default is "ncomp1st" (this is version 3.0 or later).

eta: numeric scalar indicating the parameter indexing the penalty family. This version contains only choice 1.

type: a character, indicating the penalty family. In this version, only one choice is available: "lasso."

inX: a vector or list of numeric vectors specifying the variables in X, always included in the model.

inY: a vector or list of numeric vectors specifying the variables in Y, always included in the model.

muX: a numeric scalar for the weight of X for the supervised case. 0 <= muX <= 1.

muY: a numeric scalar for the weight of Y for the supervised case. 0 <= muY <= 1.

comp: numeric scalar for the number of components to be considered or the maximum candidate number of components.

nfold: number of folds - default is 5.

maxrep: numeric scalar for the number of iteration.

minpct: minimum candidate parameters defined as a percentile of automatically determined (possible) candidates.

maxpct: maximum candidate parameters defined as a percentile of automatically determined (possible) candidates.

maxpct4ncomp: maximum candidate parameters defined as a percentile of automatically determined (possible) candidates.

criterion: a character, the evaluation criterion, "CV" for cross-validation, based on a matrix element-wise error, and "BIC" for Bayesian information criteria. The "BIC" is the default.

criterion4ncomp: a character, the evaluation criterion for the selection of the number of components, "CV" for cross-validation, based on a matrix element-wise error, and "BIC" for Bayesian information criteria.

whichselect: which blocks selected.

homo: same parameters.

intseed: seed number for the random number in the parameter estimation algorithm.

x: an object of class "optparasearch", usually, a result of a call to optparasearch.

...: further arguments passed to or from other methods.
Details

A function for identifying the regularized sparseness parameters lambdaX and lambdaY and the number of components for msma. Four search methods are available. The "simultaneous" method identifies the number of components by searching the regularized parameters in each component. The "regpara1st" identifies the regularized parameters by fixing the number of components, then searching for the number of components with the selected regularized parameters. The "ncomp1st" method identifies the number of components with a regularized parameter of 0, then searches for the regularized parameters with the selected number of components. The "regparaonly" method searches for the regularized parameters with a fixed number of components.

Value

- optncomp: Optimal number of components
- optlambdaX: Optimal parameters for X
- optlambdaY: Optimal parameters for Y
- mincriterion: Minimum criterion value
- criteria: All resulting criterion values in the process
- pararange: Range of candidates parameters

Examples

```r
##### data #####
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y

##### Regularized parameters search #####
opt1 = optparasearch(X, Y, search.method = "regparaonly", comp=1, nfold=5, maxrep=2)
opt1
fit4 = msma(X, Y, comp=opt1$optncomp, lambdaX=opt1$optlambdaX, lambdaY=opt1$optlambdaY)
fit4
summary(fit4)

##### Restrict search range #####
opt2 = optparasearch(X, Y, comp=3, nfold=5, maxrep=2, minpct=0.5)
opt2
```

Description

plot method for class "msma".
Usage

```r
## S3 method for class 'msma'
plot(
  x,
  v = c("weight", "score", "cpev")[1],
  axes = 1,
  axes2 = 1,
  block = c("block", "super")[1],
  plottype = c("bar", "scatter")[1],
  XY = c("X", "Y", "XY")[1],
  col = NULL,
  signflip = FALSE,
  xlim = NULL,
  ylim = NULL,
  ...)
```

Arguments

- `x`: an object of class "msma." Usually, a result of a call to `msma`.
- `v`: a character, "weight" for the weight, "score" for the score, and "cpev" for the cumulative percentage of explained variance (CPEV).
- `axes`: a numeric (or vector), specifying the root component(s) to plot.
- `axes2`: a numeric (or vector), specifying the nested component(s) to plot.
- `block`: a character, indicating which the "block" or "super" is used.
- `plottype`: a character, indicating the plot type. "bar" for the bar plot, "scatter" for the scatter plot.
- `XY`: a character, indicating "X" or "Y". "XY" for the scatter plots using X and Y scores from PLS.
- `col`: a color vector.
- `signflip`: a numeric vector if the sign in the block is flipped to pose the super as positive.
- `xlim`: a numeric vector x coordinate ranges.
- `ylim`: a numeric vector y coordinate ranges.
- `...`: further arguments passed to or from other methods.

Details

This function provides a plot of results.

Examples

```r
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y
fit1 = msma(X, Y, comp=1, lambdaX=2, lambdaY=1:3)
#plot(fit1)
```
predict.msma

**Description**

predict method for class "msma".

**Usage**

```r
## S3 method for class 'msma'
predict(object, newX, newY = NULL, ...)
```

**Arguments**

- `object` an object of class "msma", usually, a result of a call to `msma`
- `newX` a matrix in which to look for variables with which to predict X.
- `newY` a matrix in which to look for variables with which to predict Y.
- `...` further arguments passed to or from other methods.

**Details**

This function produces a prediction from new data based on `msma` fit. It is mainly used in cross-validation.

**Value**

- `X` predicted X
- `sbX` block score for X
- `Y` predicted Y
- `sbY` block score for Y

**Examples**

```r
##### data #####
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = tmpdata$X; Y = tmpdata$Y

##### Two Component #####
fit2 = msma(X, Y, comp=2, lambdaX=2, lambdaY=1:3)
summary(fit2)

##### Predict #####
test = predict(fit2, newX=X, newY=Y)
```
Regularized Parameters Search

Description
Regularized parameters search method for "msma".

Usage
regparasearch(
  X,
  Y = NULL,
  Z = NULL,
  eta = 1,
  type = "lasso",
  inX = NULL,
  inY = NULL,
  inXsup = NULL,
  inYsup = NULL,
  muX = 0,
  muY = 0,
  comp = 1,
  nfold = 5,
  maxrep = 3,
  minpct = 0,
  maxpct = 1,
  criterion = c("CV", "BIC")[1],
  whichselect = NULL,
  homo = NULL,
  intseed = 1
)

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

Arguments
X a matrix or list of matrices indicating the explanatory variable(s). This parameter is required.
Y a matrix or list of matrices indicating objective variable(s). This is optional. If there is no input for Y, then PCA is implemented.
Z a vector, response variable(s) for implementing the supervised version of (multi-block) PCA or PLS. This is optional. The length of Z is the number of subjects. If there is no input for Z, then unsupervised PLS/PCA is implemented.
eta numeric scalar indicating the parameter indexing the penalty family. This version contains only choice 1.
type a character, indicating the penalty family. In this version, only one choice is available: "lasso."

inX a (list of) numeric vector to specify the variables of X which are always in the model.
inY a (list of) numeric vector to specify the variables of X which are always in the model.
inXsup a (list of) numeric vector to specify the blocks of X which are always in the model.
inYsup a (list of) numeric vector to specify the blocks of Y which are always in the model.

muX a numeric scalar for the weight of X for the supervised case. 0 <= muX <= 1.

muY a numeric scalar for the weight of Y for the supervised case. 0 <= muY <= 1.
comp numeric scalar for the maximum number of components to be considered.
nfold number of folds. Default is 5.
maxrep numeric scalar for the number of iteration.
minpct percent of minimum candidate parameters.
maxpct percent of maximum candidate parameters.
criterion a character, the evaluation criterion, "CV" for cross-validation, based on a matrix element-wise error, and "BIC" for Bayesian information criteria. The "BIC" is the default.
whichselect which blocks selected.
homo same parameters.
inseed seed number for the random number in the parameter estimation algorithm.

Details
This is a function for identifying the regularized parameters of sparseness lambdaX and lambdaY for msma. The initial range of candidates is computed based on fit, with regularized parameter values of 0. A binary search is conducted for dividing the parameter range into two regions. The representative value for the region is a median value, and the optimal region is selected using the minimum criteria obtained from the fit with that median value. The CV error or BIC can be used as criteria. The selected region is also divided into two region and the same process is iterated by maxrep times. Thus, the final median value in the selected region is set to be the optimal regularized parameter. The search is conducted with combinations of parameters for X and Y. The range of candidates for regularized parameters can be restricted, with a percentile of the limit (minimum or maximum) for the range.

Value

optlambdaX Optimal parameters for X
optlambdaY Optimal parameters for Y
Examples

```r
### data ###
tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = c(20, 15), seed=1)
X = tmpdata$X; Y = tmpdata$Y

### Regularized parameters search ###
opt1 = regparasearch(X, Y, comp=1, criterion="BIC", maxrep=2,
                      whichselect=c("X", "Y", "Xsup", "Ysup"))
fit4 = msma(X, Y, comp=1, lambdaX=opt1$optlambdaX, lambdaY=opt1$optlambdaY,
            lambdaXsup=opt1$optlambdaXsup, lambdaYsup=opt1$optlambdaYsup)
fit4
summary(fit4)
```

---

**simdata**

*Simulate Data sets*

**Description**

This is a function for generating multiblock data based on the multivariable normal distribution

**Usage**

`simdata(n = 100, rho = 0.8, Yps = c(100, 120, 150), Xps = 500, seed = 1)`

**Arguments**

- `n`: a numeric scalar, sample size.
- `rho`: a numeric scalar, correlation coefficient.
- `Yps`: a numeric vector, numbers of columns for Y. The length of vector corresponds to the number of blocks.
- `Xps`: a numeric vector, numbers of columns for X. The length of vector corresponds to the number of blocks.
- `seed`: a seed number for generating random numbers.

**Details**

The output is a list of matrices.

**Value**

- `X`: Simulated X which has a list form
- `Y`: Simulated Y which has a list form
strsimdata

Structured Simulate Data sets

Description

This is a function for generating multiblock data based on the multivariable normal distribution

Usage

strsimdata(
  n = 100,
  WX = NULL,
  ncomp = 5,
  Xps = 10,
  Yps = FALSE,
  rho = 0.8,
  Ztype = c("none", "binary", "prob")[1],
  cz = c(1, 1),
  cwx = c(0.1, 0.1),
  cwy = c(0.1, 0.1),
  seed = 1,
  minpct = 0.25,
  maxpct = 0.75
)

Arguments

n                        a numeric scalar, sample size.
WX                       a matrix or a list, weights.
ncomp                    number of components
Xps                      a numeric vector, numbers of columns for X. The length of vector corresponds to the number of blocks.
Yps                      a numeric vector, numbers of columns for Y. The length of vector corresponds to the number of blocks.
rho                     a numeric, correlation
Ztype                    a character, outcome type ("none", "binary", "prob").
 cz                       a numeric vector, scale for outcome
cwx                      a numeric vector, scale for weights of X
cwy                      a numeric vector, scale for weights of Y
seed                     a seed number for generating random numbers.
minpct                   minimum percent of nonzero
maxpct                   maximum percent of nonzero
Details

The output is a list of matrices.

Value

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Simulated X which has a list form</td>
</tr>
<tr>
<td>Y</td>
<td>Simulated Y which has a list form</td>
</tr>
<tr>
<td>Z</td>
<td>Simulated Z which has a vector form</td>
</tr>
<tr>
<td>ncomp</td>
<td></td>
</tr>
<tr>
<td>Xps</td>
<td></td>
</tr>
<tr>
<td>nZeroX</td>
<td></td>
</tr>
<tr>
<td>idxZeroX</td>
<td></td>
</tr>
<tr>
<td>Yps</td>
<td></td>
</tr>
<tr>
<td>nZeroY</td>
<td></td>
</tr>
<tr>
<td>idxZeroY</td>
<td></td>
</tr>
<tr>
<td>WX</td>
<td></td>
</tr>
<tr>
<td>WY</td>
<td></td>
</tr>
<tr>
<td>ZcoefX</td>
<td></td>
</tr>
<tr>
<td>ZcoefY</td>
<td></td>
</tr>
</tbody>
</table>

summary.msma

Summarizing Fits

Description

summary method for class "msma".

Usage

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

## S3 method for class 'summary.msma'
print(x, ...)

Arguments

object, x  an object of class "msma", usually, a result of a call to msma

...  further arguments passed to or from other methods.

Details

This function provide the summary of results.
Examples

##### data #####
Tmpdata = simdata(n = 50, rho = 0.8, Yps = c(10, 12, 15), Xps = 20, seed=1)
X = Tmpdata$X; Y = Tmpdata$Y

##### One Component #####
Fit1 = msma(X, Y, comp=1, lambdaX=2, lambdaY=1:3)
summary(fit1)
Index

* documentation
  msma-package, 2
* print
  summary.msma, 20

cvmmsa, 2
hcmsma, 4
msma, 2, 4, 5, 5, 7, 14, 15, 20
msma-package, 2
ncompsearch, 8, 10
optparasearch, 11
plot.msma, 13
plot.ncompsearch (ncompsearch), 8
predict.msma, 15
print.msma (msma), 5
print.ncompsearch (ncompsearch), 8
print.optparasearch (optparasearch), 11
print.regparasearch (regparasearch), 16
print.summary.msma (summary.msma), 20
regparasearch, 16, 17
simdata, 18
strsimdata, 19
summary.msma, 20