

Package: o2plsda (via r-universe)

October 10, 2026

Type Package

Title Multiomics Data Integration

Version 0.0.28

Description Provides functions to do 'O2PLS-DA' analysis for multiple omics data integration. The algorithm came from ``O2-PLS, a two-block ($X \pm Y$) latent variable regression (LVR) method with an integral OSC filter" which published by Johan Trygg and Svante Wold at 2003 <doi:10.1002/cem.775>. 'O2PLS' is a bidirectional multivariate regression method that aims to separate the covariance between two data sets (it was recently extended to multiple data sets) (Löfstedt and Trygg, 2011 <doi:10.1002/cem.1388>; Löfstedt et al., 2012 <doi:10.1016/j.aca.2013.06.026>) from the systematic sources of variance being specific for each data set separately.

License GPL-3 + file LICENSE

Imports dplyr, magrittr, parallel, ggplot2, ggrepel, methods, stats, grid, gridExtra, igraph, plotly, reshape2

Encoding UTF-8

Suggests knitr, markdown, rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.3

URL <https://github.com/guokai8/o2plsda>

NeedsCompilation yes

Author Kai Guo [aut, cre], Junguk Hur [aut], Eva Feldman [aut]

Maintainer Kai Guo <guokai8@gmail.com>

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libuv1-dev libxml2-dev libssl-dev

Repository <https://guokai8.r-universe.dev>

Date/Publication 2025-12-02 01:32:34 UTC

RemoteUrl <https://github.com/guokai8/o2plsda>

RemoteRef HEAD

RemoteSha 56d3024a3c6ecae31f7c6cab7b11c6bc1b1098d2

Contents

\$,sparse_plsda-method	3
\$.SparseO2pls-method	3
apply_sparsity	4
calculate_mahalanobis_distances	4
calculate_prediction_probabilities	5
compare_models	5
loadings	6
loadings.o2plsda	7
loadings.plsda	7
o2cv	8
o2cv_sparse	9
o2pls	10
O2pls-class	11
oplsda	12
plot.O2pls	13
plot.o2plsda	14
plot.plsda	16
plot.sparse_plsda	17
plot.SparseO2pls	18
plot.stability_selection	21
plot.TuneResult	22
plsda	23
predict,sparse_plsda-method	24
predict,SparseO2pls-method	25
predict.sparse_plsda	25
predict.SparseO2pls	26
print.O2pls	26
print.plsda	27
print.SparseO2pls	27
scad_threshold	28
scores	28
scores.O2pls	29
scores.o2plsda	29
scores.plsda	30
selected_var_names	30
selected_vars	31
selected_vars.default	31
selected_vars.sparse_plsda	32
selected_vars.SparseO2pls	32
sparse_o2pls	33
sparse_plsda	36
SparseO2pls-class	39

<code>\$,sparse_plsda-method</code>	3
sparsity_info	39
stability_selection	40
summary.O2pls	43
summary.plsda	44
summary.sparse_plsda	45
summary.SparseO2pls	45
tune_sparse_keepX	46
tune_sparse_o2pls	47
TuneResult-class	47
vip	48
Index	49

<code>\$,sparse_plsda-method</code>	<i>Extract elements from sparse_plsda objects</i>
-------------------------------------	---

Description

Extract elements from sparse_plsda objects

Usage

```
## S4 method for signature 'sparse_plsda'
x$name
```

Arguments

x	A sparse_plsda object
name	Character string specifying the element to extract

Value

The requested element from the sparse_plsda object

<code>\$,SparseO2pls-method</code>	<i>Extract elements from SparseO2pls objects</i>
------------------------------------	--

Description

Extract elements from SparseO2pls objects

Usage

```
## S4 method for signature 'SparseO2pls'
x$name
```

Arguments

x	A SparseO2pls object
name	Character string specifying the element to extract

Value

The requested element from the SparseO2pls object

apply_sparsity	<i>Enhanced sparsity application function (FIXED VERSION)</i>
----------------	---

Description

Enhanced sparsity application function (FIXED VERSION)

Usage

```
apply_sparsity(x, keep_n, lambda = 0.1, penalty = "lasso")
```

Arguments

x	Vector of values to apply sparsity to
keep_n	Number of variables to keep
lambda	Regularization parameter for penalty
penalty	Type of penalty ("lasso" or "ridge")

calculate_mahalanobis_distances	<i>Calculate Mahalanobis distances for classification</i>
---------------------------------	---

Description

This function computes Mahalanobis distances between score vectors and class centroids.

Usage

```
calculate_mahalanobis_distances(scores, classification_info)
```

Arguments

scores	Matrix of score vectors (samples x components)
classification_info	List containing centroids and other classification information

Value

Matrix of distances (samples x classes)

`calculate_prediction_probabilities`*Calculate prediction probabilities from distances*

Description

This function converts distance matrices to classification probabilities using softmax transformation.

Usage

```
calculate_prediction_probabilities(distances)
```

Arguments

`distances` Matrix of distances (samples x classes)

Value

Matrix of probabilities (samples x classes)

`compare_models`*Compare multiple O2PLS models*

Description

Compare multiple O2PLS models

Usage

```
compare_models(..., type = "performance", names = NULL)
```

Arguments

`...` Multiple O2PLS model objects to compare
`type` Character. Type of comparison: "performance", "summary"
`names` Character vector. Names for the models (optional)

Value

ggplot object showing model comparison

Examples

```
## Not run:
# Fit regular and sparse models
regular_fit <- o2pls(X, Y, nc = 2, nx = 1, ny = 1)
sparse_fit <- sparse_o2pls(X, Y, nc = 2, nx = 1, ny = 1, keepX = c(40, 30))

# Compare models
compare_models(regular_fit, sparse_fit, type = "performance")

## End(Not run)
```

loadings	<i>Extract the loadings from an O2PLS fit</i>
----------	---

Description

This function extracts loading parameters from an O2PLS fit
This function extracts loading parameters from an O2PLS fit

Usage

```
loadings(x, ...)
```

S3 method for class 'O2pls'

```
loadings(x, loading = c("Xjoint", "Yjoint", "Xorth", "Yorth"), ...)
```

Arguments

x	Object of class O2pls
...	For consistency
loading	the loadings for one of "Xjoint", "Yjoint", "Xorth", "Yorth"

Value

Loading matrix
Loading matrix

loadings.o2plsda	<i>extract the loading value from the O2PLSDA analysis</i>
------------------	--

Description

extract the loading value from the O2PLSDA analysis

Usage

```
## S3 method for class 'o2plsda'  
loadings(x, loading = "Xloading", ...)
```

Arguments

x	Object of class o2plsda
loading	the loadings for one of "Xjoint", "Yjoint", "Xorth", "Yorth"
...	For consistency

loadings.plsda	<i>extract the loading value from the PLSDA analysis</i>
----------------	--

Description

extract the loading value from the PLSDA analysis

Usage

```
## S3 method for class 'plsda'  
loadings(x, ...)
```

Arguments

x	Object of class plsda
...	For consistency

`o2cv`*Cross validation for O2PLS*

Description

Cross validation for O2PLS

Usage

```
o2cv(  
  X,  
  Y,  
  nc,  
  nx,  
  ny,  
  group = NULL,  
  nr_folds = 5,  
  ncores = 1,  
  scale = FALSE,  
  center = FALSE  
)
```

Arguments

<code>X</code>	a Numeric matrix (input)
<code>Y</code>	a Numeric matrix (input)
<code>nc</code>	Integer. Number of joint PLS components.
<code>nx</code>	Integer. Number of orthogonal components in X
<code>ny</code>	Integer. Number of orthogonal components in Y
<code>group</code>	a vector to indicate the group for Y
<code>nr_folds</code>	Integer to indicate the folds for cross validation
<code>ncores</code>	Integer. Number of CPUs to use for cross validation
<code>scale</code>	boolean values determining if data should be scaled or not
<code>center</code>	boolean values determining if data should be centered or not

Value

a data frame with the Q and RMSE values

Author(s)

Kai Guo

Examples

```

set.seed(123)
X = matrix(rnorm(500),50,10)
Y = matrix(rnorm(500),50,10)
X = scale(X, scale = TRUE)
Y = scale(Y, scale = TRUE)
# group factor could be omitted if you don't have any group
group <- rep(c("Ctrl","Treat"), each = 25)
cv <- o2cv(X, Y, 1:2, 1:2, 1:2, group=group, nr_folds = 2, ncores=1)

```

o2cv_sparse

*Enhanced sparse cross-validation that integrates with existing o2cv***Description**

Enhanced sparse cross-validation that integrates with existing o2cv

Usage

```

o2cv_sparse(
  X,
  Y,
  nc,
  nx,
  ny,
  keepX_range = seq(10, 50, 10),
  keepY_range = seq(10, 50, 10),
  group = NULL,
  nr_folds = 5,
  ncores = 1,
  validation = "Mfold"
)

```

Arguments

X	Input data matrix (samples x variables)
Y	Response data matrix (samples x variables)
nc	Number of joint components
nx	Number of X-specific orthogonal components
ny	Number of Y-specific orthogonal components
keepX_range	Range of values for number of X variables to keep
keepY_range	Range of values for number of Y variables to keep
group	Grouping variable for stratified CV
nr_folds	Number of cross-validation folds
ncores	Number of cores for parallel processing
validation	Cross-validation method ("Mfold", "loo", "bootstrap")

o2pls	<i>fit O2PLS model with best nc, nx, ny</i>
-------	---

Description

fit O2PLS model with best nc, nx, ny

Usage

```
o2pls(X, Y, nc, nx, ny, scale = FALSE, center = FALSE)
```

Arguments

X	a Numeric matrix (input)
Y	a Numeric matrix (input)
nc	Integer. Number of joint PLS components.
nx	Integer. Number of orthogonal components in X
ny	Integer. Number of orthogonal components in Y
scale	boolean values determining if data should be scaled or not
center	boolean values determining if data should be centered or not

Value

An object containing

Xscore	Joint X scores
Xloading	Joint X loadings
Yscore	Joint Y scores
Yloading	Joint Y loadings
TYosc	Orthogonal X scores
PYosc	Orthogonal X loadings
WYosc	Orthogonal X weights
UXosc	Orthogonal Y scores
PXosc	Orthogonal Y loadings
CXosc	Orthogonal Y weights
BU	Regression coefficient in $Tt \sim U$
BT	Regression coefficient in $U \sim Tt$
Xhat	Prediction of X with Y
Yhat	Prediction of Y with X
R2Xhat	Variation of the predicted X as proportion of variation in X
R2Yhat	Variation of the predicted Y as proportion of variation in Y

R2X	Variation of the modeled part in X (defined by Joint + Orthogonal variation) as proportion of total variation in X
R2Y	Variation of the modeled part in Y (defined by Joint + Orthogonal variation) as proportion of total variation in Y
R2Xcorr	Variation of the joint part in X
R2Ycorr	Variation of the joint part in Y
R2Xo	Variation of the orthogonal part in X as proportion of variation in X
R2Yo	Variation of the orthogonal part in Y as proportion of variation in Y
R2Xp	Variation in X joint part predicted by Y Joint part
R2Yp	Variation in Y joint part predicted by X Joint part
varXj	Variation in each Latent Variable (LV) in X Joint part
varYj	Variation in each Latent Variable (LV) in Y Joint part
varXorth	Variation in each Latent Variable (LV) in X Orthogonal part
varYorth	Variation in each Latent Variable (LV) in Y Orthogonal part
Exy	Residuals in X
Fxy	Residuals in Y

Author(s)

Kai Guo

Examples

```

set.seed(123)
X = matrix(rnorm(500),50,10)
Y = matrix(rnorm(500),50,10)
X = scale(X, scale = TRUE)
Y = scale(Y, scale = TRUE)
fit <- o2pls(X, Y, 1, 2, 2)
summary(fit)

```

O2pls-class

*Class "O2pls" This class represents the Annotation information***Description**

Class "O2pls" This class represents the Annotation information

Slots

X a Numeric matrix (input)
Y a Numeric matrix (input)
params paramaters ysed in o2pls analysis
results list of o2pls results

Author(s)

Kai Guo

oplsda

*Orthogonal partial least squares discriminant analysis***Description**

Computes orthogonal scores partial least squares regressions with the NIPALS algorithm. It return a comprehensive set of pls outputs (e.g. scores and vip).

Usage

```
oplsda(X, Y, nc, scale = FALSE, center = TRUE, maxiter = 100, tol = 1e-05)
```

Arguments

X	a O2pls object or a matrix of predictor variables.
Y	a single vector indicate the group
nc	the number of pls components (the one joint components + number of orthogonal components).
scale	logical indicating whether X must be scaled (suggest TRUE).
center	boolean values determining if data should be centered or not
maxiter	maximum number of iterations.
tol	limit for convergence of the algorithm in the nipals algorithm.

Value

a list containing the following elements:

- nc the number of components used(one joint components + number of orthogonal components
- scores a matrix of scores corresponding to the observations in X, The components retrieved correspond to the ones optimized or specified.
- Xloadings a matrix of loadings corresponding to the explanatory variables. The components retrieved correspond to the ones optimized or specified.
- Yloadings a matrix of partial least squares loadings corresponding to Y
- vip the VIP matrix.
- xvar a matrix indicating the standard deviation of each component (sd), the variance explained by each single component (explained_var) and the cumulative explained variance (cumulative_explained_var). These values are computed based on the data used to create the projection matrices.
- projection_matrix the matrix of projection matrix
- weight a matrix of partial least squares ("pls") weights.

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(50),10,5)
Y <- matrix(rnorm(50),10,5)
fit <- o2pls(X,Y,2,1,1)
yy <- rep(c(0,1),5)
fit0 <- oplstda(fit,yy,2)
```

plot.O2pls

*Score or loading plot for the O2PLS results***Description**

Score or loading plot for the O2PLS results

Usage

```
## S3 method for class 'O2pls'
plot(
  x,
  type = "score",
  var = "Xjoint",
  group = NULL,
  ind = c(1, 2),
  color = NULL,
  top = 20,
  ellipse = FALSE,
  order = FALSE,
  pt.size = 3,
  label = TRUE,
  label.size = 4,
  repel = TRUE,
  rotation = FALSE,
  ...
)
```

Arguments

x	an O2pls object
type	score or loading
var	specify Xjoint
group	color used for score plot
ind	which components to be used for score plot or loading plot

color	color used for score or loading plot
top	the number of largest loading value to plot
ellipse	TRUE/FALSE
order	order by the value or not
pt.size	point size
label	plot label or not (TRUE/FALSE)
label.size	label size
repel	use ggrepel to show the label or not
rotation	flip the figure or not (TRUE/FALSE)
...	For consistency

Value

a ggplot2 object

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(50),10,5)
Y <- matrix(rnorm(50),10,5)
fit <- o2pls(X,Y,2,1,1)
plot(fit, type="score")
```

plot.o2plsda	<i>Score, VIP or loading plot for the O2PLS results</i>
--------------	---

Description

Score, VIP or loading plot for the O2PLS results

Usage

```
## S3 method for class 'o2plsda'
plot(
  x,
  type = "score",
  group = NULL,
  ind = c(1, 2),
  color = NULL,
  top = 20,
  ellipse = FALSE,
  order = FALSE,
```

```

    pt.size = 3,
    label = TRUE,
    label.size = 4,
    repel = FALSE,
    rotation = FALSE,
    ...
)

```

Arguments

x	an o2plsda object
type	score, vip or loading
group	color used for score plot
ind	which components to be used for score plot or loading plot
color	color used for score or loading plot
top	the number of largest loading value to plot
ellipse	TRUE/FALSE
order	order by the value or not
pt.size	point size
label	plot label or not (TRUE/FALSE)
label.size	label size
repel	use ggrepel to show the label or not
rotation	flip the figure or not (TRUE/FALSE)
...	For consistency

Value

a ggplot2 object

Author(s)

Kai Guo

Examples

```

X <- matrix(rnorm(50),10,5)
Y <- matrix(rnorm(50),10,5)
fit <- o2pls(X,Y,2,1,1)
yy <- rep(c(0,1),5)
fit0 <- oplpsda(fit,yy,2)
plot(fit0, type="score", group = factor(yy))

```

plot.plsda

*Score, VIP or loading plot for the plsda results***Description**

Score, VIP or loading plot for the plsda results

Usage

```
## S3 method for class 'plsda'
plot(
  x,
  type = "score",
  group = NULL,
  ind = c(1, 2),
  color = NULL,
  top = 20,
  ellipse = FALSE,
  order = FALSE,
  pt.size = 3,
  label = TRUE,
  label.size = 4,
  repel = FALSE,
  rotation = FALSE,
  ...
)
```

Arguments

x	an plsda object
type	score, vip or loading
group	color used for score plot
ind	which components to be used for score plot or loading plot
color	color used for score or loading plot
top	the number of largest loading value to plot
ellipse	TRUE/FALSE
order	order by the value or not
pt.size	point size
label	plot label or not (TRUE/FALSE)
label.size	label size
repel	use ggrepel to show the label or not
rotation	flip the figure or not (TRUE/FALSE)
...	For consistency

Value

a ggplot2 object

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(500),10,50)
Y <- rep(c("a","b"),each=5)
fit0 <- plsda(X,Y,2)
plot(fit0, type = "score", group = factor(Y))
```

plot.sparse_plsda	<i>Plot method for Sparse PLS-DA results</i>
-------------------	--

Description

Plot method for Sparse PLS-DA results

Usage

```
## S3 method for class 'sparse_plsda'
plot(
  x,
  type = "score",
  component = c(1, 2),
  group = NULL,
  top = 20,
  color = NULL,
  ellipse = FALSE,
  ...
)
```

Arguments

x	A sparse_plsda object (corrected class name)
type	Character. Type of plot: "score", "loading", "vip", "selection"
component	Integer or vector. Which component(s) to plot
group	Factor. Grouping variable (usually the Y classes)
top	Integer. Number of top variables to display
color	Custom colors for groups
ellipse	Logical. Add confidence ellipses to score plots
...	Additional plotting parameters

Examples

```
# Generate example data
set.seed(123)
n <- 100
p <- 150

X <- matrix(rnorm(n * p), n, p)
# Add class-specific signals
classes <- factor(rep(c("A", "B", "C"), length.out = n))
X[classes == "A", 1:20] <- X[classes == "A", 1:20] + 1.5
X[classes == "B", 21:40] <- X[classes == "B", 21:40] + 1.5
X[classes == "C", 41:60] <- X[classes == "C", 41:60] + 1.5

colnames(X) <- paste0("Gene_", 1:p)

# Fit sparse PLS-DA (using corrected class name)
sparse_plsda_fit <- sparse_plsda(X, classes, nc = 2, keepX = c(30, 25))
class(sparse_plsda_fit) <- "sparse_plsda" # Ensure correct class

# Score plot
plot(sparse_plsda_fit, type = "score", group = classes, ellipse = FALSE)

# Loading plot
plot(sparse_plsda_fit, type = "loading", component = 1, top = 20)

# VIP plot
plot(sparse_plsda_fit, type = "vip", component = 1, top = 25)
```

plot.SparseO2pls

Plot method for SparseO2pls S4 objects

Description

Creates various plots for sparse O2PLS models stored as S4 objects, including score plots, loading plots, sparsity patterns, and variable selection summaries.

Usage

```
## S3 method for class 'SparseO2pls'
plot(
  x,
  type = "score",
  component = c(1, 2),
  block = "X",
  group = NULL,
  top = 20,
  threshold = 1e-10,
  color = NULL,
```

```

    ellipse = FALSE,
    title = NULL,
    ...
)

```

Arguments

<code>x</code>	A SparseO2pls S4 object
<code>type</code>	Character. Type of plot: "score", "loading", "sparsity", "selection", "biplot", "contribution", "comparison", or "diagnostic"
<code>component</code>	Integer or vector. Which component(s) to plot (default: c(1,2))
<code>block</code>	Character. Which data block: "X", "Y", or "both" (default: "X")
<code>group</code>	Factor. Grouping variable for score plots
<code>top</code>	Integer. Number of top variables to show in loading plots (default: 20)
<code>threshold</code>	Numeric. Threshold for variable selection display (default: 1e-10)
<code>color</code>	Character vector. Custom colors for groups
<code>ellipse</code>	Logical. Add confidence ellipses to score plots (default: TRUE)
<code>title</code>	Character. Custom plot title
<code>...</code>	Additional plotting parameters

Value

ggplot2 object or plotly object (for 3D plots)

Examples

```

# Generate example data for SparseO2pls plotting
set.seed(456)
n <- 120
p_X <- 180
p_Y <- 120

# Create structured data
X <- matrix(rnorm(n * p_X), n, p_X)
Y <- matrix(rnorm(n * p_Y), n, p_Y)

# Add correlated signal
signal <- matrix(rnorm(n * 4), n, 4)
X[, 1:40] <- signal %% matrix(rnorm(4 * 40), 4, 40) +
  matrix(rnorm(n * 40, sd = 0.4), n, 40)
Y[, 1:30] <- signal %% matrix(rnorm(4 * 30), 4, 30) +
  matrix(rnorm(n * 30, sd = 0.4), n, 30)

# Add meaningful names
colnames(X) <- paste0("Gene_", 1:p_X)
colnames(Y) <- paste0("Protein_", 1:p_Y)
rownames(X) <- rownames(Y) <- paste0("Sample_", 1:n)

```

```

# Create grouping variable
treatment_groups <- factor(rep(c("Control", "Low_Dose", "High_Dose", "Recovery"),
                              each = n/4))

# Fit sparse O2PLS and convert to S4 object
sparse_fit_list <- sparse_o2pls(X, Y, nc = 3, nx = 1, ny = 1,
                              keepX = c(50, 40, 30), keepY = c(35, 25, 20))

# Convert to S4 SparseO2pls object
sparse_s4 <- new("SparseO2pls",
                 X = X, Y = Y,
                 params = sparse_fit_list$params,
                 results = sparse_fit_list$results,
                 sparsity = sparse_fit_list$sparsity)

# =====
# SCORE PLOTS
# =====

# 2D score plot for X block with treatment groups
p1 <- plot(sparse_s4, type = "score", block = "X",
           component = c(1, 2), group = treatment_groups, ellipse = FALSE)
print(p1)

# 2D score plot for Y block
p2 <- plot(sparse_s4, type = "score", block = "Y",
           component = c(1, 3), group = treatment_groups,
           color = c("#E31A1C", "#1F78B4", "#33A02C", "#FF7F00"))
print(p2)

# 3D score plot
p3 <- plot(sparse_s4, type = "score", block = "X",
           component = c(1, 2, 3), group = treatment_groups)
print(p3)

# =====
# LOADING PLOTS
# =====

# Top gene loadings for component 1
p4 <- plot(sparse_s4, type = "loading", block = "X",
           component = 1, top = 30)
print(p4)

# Top protein loadings for component 2
p5 <- plot(sparse_s4, type = "loading", block = "Y",
           component = 2, top = 25,
           title = "Top Protein Loadings - Component 2")
print(p5)

# Multi-component loading comparison
p6 <- plot(sparse_s4, type = "loading", block = "X",
           component = c(1, 2, 3), top = 35)

```

```

print(p6)

# =====
# SPARSITY AND SELECTION PLOTS
# =====

# Sparsity pattern for both blocks
p7 <- plot(sparse_s4, type = "sparsity", block = "both")
print(p7)

# Variable selection summary
p8 <- plot(sparse_s4, type = "selection")
print(p8)

# =====
# ADVANCED PLOTS
# =====

# Biplot combining scores and loadings
p9 <- plot(sparse_s4, type = "biplot",
           component = c(1, 2), group = treatment_groups, top = 20)
print(p9)

# Variable contribution analysis
p10 <- plot(sparse_s4, type = "contribution", block = "X",
            component = 1, top = 25)
print(p10)

# Diagnostic plots
p11 <- plot(sparse_s4, type = "diagnostic")
print(p11)

```

plot.stability_selection

Enhanced plot method for stability_selection objects

Description

Enhanced plot method for stability_selection objects

Usage

```
## S3 method for class 'stability_selection'
plot(x, component = 1, type = "heatmap", top_n = 50, ...)
```

Arguments

x A stability_selection object

component	Integer. Which component to visualize
type	Character. Type of plot: "heatmap", "barplot", "threshold", "summary"
top_n	Integer. Number of top variables to show
...	Additional plotting parameters

Examples

```
## Not run:
# Assuming you have a stability_selection object
stability_res <- stability_selection(X, Y, nc = 2, n_bootstrap = 50)

# Stability heatmap
plot(stability_res, type = "heatmap", component = 1)

# Stability barplot
plot(stability_res, type = "barplot", component = 1, top_n = 30)

# Threshold analysis
plot(stability_res, type = "threshold")

## End(Not run)
```

plot.TuneResult	<i>Enhanced plot method for TuneResult objects</i>
-----------------	--

Description

Enhanced plot method for TuneResult objects

Usage

```
## S3 method for class 'TuneResult'
plot(x, type = "heatmap", metric = "mean_score", ...)
```

Arguments

x	A TuneResult object
type	Character. Type of plot: "heatmap", "line", "comparison", "optimal"
metric	Character. Performance metric to visualize
...	Additional plotting parameters

Examples

```
# Example with tuning results
## Not run:
# Assuming you have a TuneResult object from tune_o2pls()
tune_results <- tune_o2pls(X, Y, nc_range = 1:3, nx_range = 0:2, ny_range = 0:2)

# Heatmap of results
plot(tune_results, type = "heatmap")

# Line plot showing trends
plot(tune_results, type = "line")

# Comparison across parameters
plot(tune_results, type = "comparison")

## End(Not run)
```

plsda

Partial least squares discriminant analysis

Description

Perform a PLS discriminant analysis

Usage

```
plsda(
  X,
  Y,
  nc,
  scale = TRUE,
  center = TRUE,
  cv = TRUE,
  nr_folds = 5,
  tol = 1e-06,
  max_iter = 100,
  q2_threshold = 0.05
)
```

Arguments

X	a matrix of predictor variables.
Y	a single vector indicate the group
nc	the number of pls components
scale	logical indicating whether X must be scaled (suggest TRUE).
center	logical indicating whether X must be centered (suggest TRUE).

<code>cv</code>	logical indicating whether cross-validation will be performed or not (suggest TRUE).
<code>nr_folds</code>	Integer to indicate the folds for cross validation.
<code>tol</code>	Convergence tolerance (default: 1e-6)
<code>max_iter</code>	Maximum number of iterations (default: 100)
<code>q2_threshold</code>	Q2 threshold for component selection (default: 0.05)

Value

a list containing PLS-DA results

`predict,sparse_plsda-method`
Predict method for sparse_plsda objects

Description

Predict method for `sparse_plsda` objects

Usage

```
## S4 method for signature 'sparse_plsda'
predict(object, newdata, ...)
```

Arguments

<code>object</code>	A <code>sparse_plsda</code> object
<code>newdata</code>	New data matrix for prediction
<code>...</code>	Additional arguments passed to <code>predict.sparse_plsda</code>

Value

Prediction results

`predict,SparseO2pls-method`*Predict method for SparseO2pls objects*

Description

Predict method for SparseO2pls objects

Usage

```
## S4 method for signature 'SparseO2pls'  
predict(object, newdata, ...)
```

Arguments

<code>object</code>	A SparseO2pls object
<code>newdata</code>	New data matrix for prediction
<code>...</code>	Additional arguments passed to <code>predict.SparseO2pls</code>

Value

Prediction results

`predict.sparse_plsda` *Enhanced predict method for sparse PLS-DA*

Description

Enhanced predict method for sparse PLS-DA

Usage

```
predict.sparse_plsda(object, newdata, dist = "max.dist", ...)
```

Arguments

<code>object</code>	A <code>sparse_plsda</code> object
<code>newdata</code>	New data matrix for prediction
<code>dist</code>	Distance method for classification ("max.dist" or "centroids.dist")
<code>...</code>	Additional arguments (currently unused)

Value

Prediction results

predict.SparseO2pls	<i>Enhanced predict method for sparse O2PLS</i>
---------------------	---

Description

Enhanced predict method for sparse O2PLS

Usage

```
predict.SparseO2pls(object, newdata, ...)
```

Arguments

object	A SparseO2pls object
newdata	New data matrix for prediction
...	Additional arguments (currently unused)

Value

Prediction results

print.O2pls	<i>Print the summary of O2PLS results.</i>
-------------	--

Description

Print the summary of O2PLS results.

Usage

```
## S3 method for class 'O2pls'
print(x, ...)
```

Arguments

x	An O2pls object
...	For consistency

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(50),10,5)
Y <- matrix(rnorm(50),10,5)
object <- o2pls(X,Y,1,1,1)
print(object)
```

print.plsda	<i>Print the summary of plsda results.</i>
-------------	--

Description

Print the summary of plsda results.

Usage

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

Arguments

x	An plsda object
...	For consistency

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(500),10,50)
Y <- rep(c("a","b"),each=5)
fit <- plsda(X,Y,2)
print(fit)
```

print.SparseO2pls	<i>Print method for sparse O2PLS results</i>
-------------------	--

Description

Print method for sparse O2PLS results

Usage

```
## S3 method for class 'SparseO2pls'
print(x, ...)
```

Arguments

x	A SparseO2pls object to print
...	Additional arguments passed to summary method

scad_threshold	<i>Enhanced SCAD thresholding function</i>
----------------	--

Description

Enhanced SCAD thresholding function

Usage

```
scad_threshold(x, lambda, a)
```

Arguments

x	Vector of values to threshold
lambda	Regularization parameter
a	SCAD parameter (typically 3.7)

Value

Thresholded values

scores	<i>Extract the scores from an O2PLS fit</i>
--------	---

Description

This function extracts score matrices from an O2PLS fit

Usage

```
scores(x, ...)
```

Arguments

x	Object of class O2pls
...	For consistency

Value

Scores matrix

scores.O2pls	<i>Extract the scores from an O2PLS fit</i>
--------------	---

Description

This function extracts scores parameters from an O2PLS fit

Usage

```
## S3 method for class 'O2pls'  
scores(x, score = c("Xjoint", "Yjoint", "Xorth", "Yorth"), ...)
```

Arguments

x	Object of class O2pls
score	the scores matrix for one of "Xjoint", "Yjoint", "Xorth", "Yorth"
...	Other arguments

Value

score matrix

scores.o2plsda	<i>Extract the scores from an O2PLS DA analysis</i>
----------------	---

Description

Extract the scores from an O2PLS DA analysis

Usage

```
## S3 method for class 'o2plsda'  
scores(x, ...)
```

Arguments

x	Object of class o2plsda
...	Other arguments

Value

score matrix

Author(s)

Kai Guo

scores.plsda	<i>Extract the scores PLSDA analysis</i>
--------------	--

Description

Extract the scores PLSDA analysis

Usage

```
## S3 method for class 'plsda'  
scores(x, ...)
```

Arguments

x	Object of class plsda
...	Other arguments

Value

score matrix

Author(s)

Kai Guo

selected_var_names	<i>Get names of selected variables</i>
--------------------	--

Description

Get names of selected variables

Usage

```
selected_var_names(x, type = "X", ...)
```

Arguments

x	A sparse model object
type	Character. Which variables to extract: "X", "Y", or "both"
...	Additional arguments passed to selected_vars

Value

Character vector of variable names or list of name vectors

selected_vars	<i>Extract selected variables from sparse models</i>
---------------	--

Description

Extract selected variables from sparse models

Usage

```
selected_vars(x, ...)
```

Arguments

x	A sparse model object
...	Additional arguments passed to methods

Value

Vector or list of selected variable indices

selected_vars.default	<i>Default method for selected_vars</i>
-----------------------	---

Description

Default method for selected_vars

Usage

```
## Default S3 method:  
selected_vars(x, ...)
```

Arguments

x	An object
...	Additional arguments

```
selected_vars.sparse_plsda
```

Extract selected variables from sparse PLS-DA models

Description

Extract selected variables from sparse PLS-DA models

Usage

```
## S3 method for class 'sparse_plsda'
selected_vars(x, ...)
```

Arguments

x	A sparse_plsda object
...	Additional arguments (currently unused)

Value

Vector of selected variable indices

```
selected_vars.SparseO2pls
```

Extract selected variables from SparseO2pls S4 objects

Description

Extract selected variables from SparseO2pls S4 objects

Usage

```
## S3 method for class 'SparseO2pls'
selected_vars(x, type = "X", ...)
```

Arguments

x	A SparseO2pls S4 object
type	Character. Which variables to extract: "X", "Y", or "both"
...	Additional arguments (currently unused)

Value

Vector of variable indices (for "X" or "Y") or list of indices (for "both")

sparse_o2pls

*Sparse Two-way Orthogonal Partial Least Squares***Description**

Performs sparse O2PLS analysis with automatic variable selection using L1 regularization (Lasso) or other penalty methods.

Usage

```
sparse_o2pls(
  X,
  Y,
  nc,
  nx = 0,
  ny = 0,
  keepX = NULL,
  keepY = NULL,
  lambda_x = 0.1,
  lambda_y = 0.1,
  penalty = "lasso",
  scale = TRUE,
  center = TRUE,
  max_iter = 100,
  tol = 1e-06
)
```

Arguments

X	Numeric matrix (samples x variables). The predictor data matrix.
Y	Numeric matrix (samples x variables). The response data matrix.
nc	Integer. Number of joint components to extract.
nx	Integer. Number of X-specific orthogonal components (default: 0).
ny	Integer. Number of Y-specific orthogonal components (default: 0).
keepX	Vector of integers. Number of X variables to keep per component. If NULL, automatically determined based on data dimensions.
keepY	Vector of integers. Number of Y variables to keep per component. If NULL, automatically determined based on data dimensions.
lambda_x	Numeric. L1 penalty parameter for X variables (default: 0.1).
lambda_y	Numeric. L1 penalty parameter for Y variables (default: 0.1).
penalty	Character. Penalty type: "lasso" (default) or "elastic".
scale	Logical. Scale variables to unit variance (default: TRUE).
center	Logical. Center variables to zero mean (default: TRUE).
max_iter	Integer. Maximum iterations for convergence (default: 100).
tol	Numeric. Convergence tolerance (default: 1e-6).

Details

Sparse O2PLS extends traditional O2PLS by incorporating variable selection through L1 regularization. This is particularly useful for high-dimensional data where only a subset of variables are relevant for the relationship between X and Y datasets.

The keepX and keepY parameters control the level of sparsity. Smaller values lead to sparser models with fewer selected variables. If not specified, reasonable defaults are chosen based on data dimensions.

Value

sparse_o2pls object containing sparse loadings, scores, and sparsity information.

Author(s)

Kai Guo

See Also

[o2pls](#), [selected_vars](#), [stability_selection](#)

Examples

```
# Example 1: Basic sparse O2PLS with automatic parameter selection
set.seed(42)
n <- 80
p_X <- 100 # High-dimensional X
p_Y <- 60  # High-dimensional Y

# Generate sparse data with only some variables being relevant
X <- matrix(rnorm(n * p_X), n, p_X)
Y <- matrix(rnorm(n * p_Y), n, p_Y)

# Create relationships between first 20 variables
true_signal <- matrix(rnorm(n * 3), n, 3) # 3 latent factors
X[, 1:20] <- true_signal %%% matrix(rnorm(3 * 20), 3, 20) +
  matrix(rnorm(n * 20, sd = 0.5), n, 20)
Y[, 1:15] <- true_signal %%% matrix(rnorm(3 * 15), 3, 15) +
  matrix(rnorm(n * 15, sd = 0.5), n, 15)

# Add variable names
colnames(X) <- paste0("X_", 1:p_X)
colnames(Y) <- paste0("Y_", 1:p_Y)
rownames(X) <- rownames(Y) <- paste0("Sample_", 1:n)

# Fit sparse O2PLS with automatic keepX/keepY selection
sparse_fit1 <- sparse_o2pls(X, Y, nc = 2)

# View results
print(sparse_fit1)
summary(sparse_fit1)
```

```

# Extract selected variables
selected_X <- selected_vars(sparse_fit1, type = "X")
selected_Y <- selected_vars(sparse_fit1, type = "Y")
selected_X_names <- selected_var_names(sparse_fit1, type = "X")
selected_Y_names <- selected_var_names(sparse_fit1, type = "Y")

cat("Selected X variables:", length(selected_X), "out of", p_X, "\n")
cat("Selected Y variables:", length(selected_Y), "out of", p_Y, "\n")
cat("X variables:", paste(head(selected_X_names, 10), collapse = ", "), "\n")

# Example 2: Controlled sparsity levels
# Specify exact number of variables to keep
sparse_fit2 <- sparse_o2pls(
  X = X,
  Y = Y,
  nc = 2,
  nx = 1,
  ny = 1,
  keepX = c(25, 20),      # Keep 25 variables in comp 1, 20 in comp 2
  keepY = c(20, 15),      # Keep 20 variables in comp 1, 15 in comp 2
  lambda_x = 0.05,
  lambda_y = 0.05,
  penalty = "lasso"
)

# Compare sparsity levels
sparsity_info1 <- sparsity_info(sparse_fit1)
sparsity_info2 <- sparsity_info(sparse_fit2)

print("Automatic sparsity:")
print(sparsity_info1)
print("Controlled sparsity:")
print(sparsity_info2)

# Example 3: Prediction with sparse model
# Split data for validation
train_idx <- 1:60
test_idx <- 61:80

X_train <- X[train_idx, ]
Y_train <- Y[train_idx, ]
X_test <- X[test_idx, ]
Y_test <- Y[test_idx, ]

# Fit on training data
sparse_train <- sparse_o2pls(X_train, Y_train, nc = 2, keepX = c(30, 25))

# Predict on test data
Y_pred <- predict(sparse_train, X_test)

# Calculate prediction error
pred_error <- sqrt(mean((Y_test - Y_pred)^2))
cat("Prediction RMSE:", round(pred_error, 4), "\n")

```

```
# Example 4: Different penalty types
# Compare Lasso vs Elastic Net
sparse_lasso <- sparse_o2pls(X, Y, nc = 2, penalty = "lasso", lambda_x = 0.1)
sparse_elastic <- sparse_o2pls(X, Y, nc = 2, penalty = "elastic", lambda_x = 0.1)

# Compare number of selected variables
cat("Lasso selected:", length(selected_vars(sparse_lasso, "X")), "X variables\n")
cat("Elastic Net selected:", length(selected_vars(sparse_elastic, "X")), "X variables\n")
```

sparse_plsda

Sparse Partial Least Squares Discriminant Analysis

Description

Performs sparse PLS-DA for classification with automatic variable selection.

Usage

```
sparse_plsda(
  X,
  Y,
  nc,
  keepX = NULL,
  validation = "Mfold",
  folds = 5,
  test.keepX = seq(5, 50, 5),
  scale = TRUE,
  center = TRUE,
  dist = "max.dist",
  tune.keepX = is.null(keepX)
)
```

Arguments

X	Numeric matrix (samples x variables). The predictor data matrix.
Y	Factor or character vector. Class labels for each sample.
nc	Integer. Number of components to extract.
keepX	Vector of integers. Number of variables to keep per component. If NULL, automatically determined.
validation	Character. Validation method for parameter tuning: "Mfold" or "loo".
folds	Integer. Number of cross-validation folds (default: 5).
test.keepX	Vector of keepX values to test during tuning.
scale	Logical. Scale variables (default: TRUE).

center	Logical. Center variables (default: TRUE).
dist	Character. Distance metric for classification: "max.dist", "centroids.dist", or "mahalanobis.dist".
tune.keepX	Logical. Automatically tune keepX parameters (default: TRUE if keepX is NULL).

Value

sparse_plsda object containing sparse loadings, scores, VIP scores, and classification results.

Author(s)

Kai Guo

See Also

[sparse_o2pls](#), [stability_selection](#), [tune_sparse_keepX](#)

Examples

```
# Example 1: Basic sparse PLS-DA
set.seed(123)
n <- 120
p <- 200 # High-dimensional feature space

# Generate classification data
X <- matrix(rnorm(n * p), n, p)

# Create three classes with different patterns
n_per_class <- n / 3
classes <- rep(c("Class_A", "Class_B", "Class_C"), each = n_per_class)

# Add class-specific signals to subsets of variables
X[classes == "Class_A", 1:20] <- X[classes == "Class_A", 1:20] + 2
X[classes == "Class_B", 21:40] <- X[classes == "Class_B", 21:40] + 2
X[classes == "Class_C", 41:60] <- X[classes == "Class_C", 41:60] + 2

# Add some noise correlation
X[, 61:80] <- X[, 1:20] + matrix(rnorm(n * 20, sd = 0.5), n, 20)

colnames(X) <- paste0("Feature_", 1:p)
Y <- factor(classes)

# Fit sparse PLS-DA with automatic tuning
sparse_plsda_fit <- sparse_plsda(
  X = X,
  Y = Y,
  nc = 2,
  validation = "Mfold",
  folds = 5,
  test.keepX = seq(10, 50, 10), # Test different sparsity levels
```

```

    tune.keepX = TRUE
  )

  # View results
  print(sparse_plsda_fit)
  summary(sparse_plsda_fit)

  # Extract selected features
  selected_features <- selected_vars(sparse_plsda_fit)
  selected_names <- selected_var_names(sparse_plsda_fit)

  cat("Selected", length(selected_features), "out of", p, "features\n")
  cat("Selected features:", paste(head(selected_names, 15), collapse = ", "), "\n")

  # Plot results
  plot(sparse_plsda_fit, type = "score", group = Y)
  plot(sparse_plsda_fit, type = "loading", top = 20)

  # Example 2: Manual keepX specification
  sparse_plsda_manual <- sparse_plsda(
    X = X,
    Y = Y,
    nc = 3,
    keepX = c(30, 25, 20), # Decreasing sparsity per component
    tune.keepX = FALSE,
    dist = "centroids.dist"
  )

  # Compare classification performance
  cat("Auto-tuned error rate:", sparse_plsda_fit$classification$overall_error, "\n")
  cat("Manual keepX error rate:", sparse_plsda_manual$classification$overall_error, "\n")

  # Example 3: Cross-validation and prediction
  # Split data
  train_idx <- sample(1:n, 0.7 * n)
  test_idx <- setdiff(1:n, train_idx)

  X_train <- X[train_idx, ]
  Y_train <- Y[train_idx]
  X_test <- X[test_idx, ]
  Y_test <- Y[test_idx]

  # Train model
  model_train <- sparse_plsda(X_train, Y_train, nc = 2, keepX = c(25, 20))

  # Predict test set
  predictions <- predict(model_train, X_test, dist = "max.dist")

  # Evaluate predictions
  confusion_matrix <- table(Predicted = predictions$class, Actual = Y_test)
  accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)

  print(confusion_matrix)

```

```
cat("Test set accuracy:", round(accuracy, 3), "\n")
```

SparseO2pls-class	<i>Class "SparseO2pls" This class represents sparse O2PLS analysis results</i>
-------------------	--

Description

Class "SparseO2pls" This class represents sparse O2PLS analysis results

Slots

- X Numeric matrix (input X)
- Y Numeric matrix (input Y)
- params List of parameters including sparsity
- results List of sparse O2PLS results
- sparsity List of sparsity-related information
- preprocessing list of preprocessing information
- call The matched call used to generate the object

Author(s)

Kai Guo

sparsity_info	<i>Get sparsity information from sparse models</i>
---------------	--

Description

Get sparsity information from sparse models

Usage

```
sparsity_info(x, ...)  
  
sparsity_info.sparse_o2pls(x, ...)  
  
sparsity_info.sparse_plsda(x, ...)  
  
## S4 method for signature 'sparse_plsda'  
sparsity_info(x, ...)  
  
sparsity_info.SparseO2pls(x, ...)  
  
## S4 method for signature 'SparseO2pls'  
sparsity_info(x, ...)
```

Arguments

x	A sparse model object
...	Additional arguments

Value

List containing sparsity statistics

stability_selection	<i>Stability Selection for Sparse Methods</i>
---------------------	---

Description

Performs bootstrap stability selection to identify robust features across multiple bootstrap samples and sparsity levels.

Usage

```
stability_selection(
  X,
  Y,
  nc,
  keepX_range = seq(10, 100, 10),
  n_bootstrap = 100,
  threshold = 0.8,
  method = "sparse_plsda",
  parallel = TRUE,
  ncores = NULL
)
```

Arguments

X	Numeric matrix (samples x variables). The predictor data matrix.
Y	Factor, character vector, or numeric matrix. Response data.
nc	Integer. Number of components to extract.
keepX_range	Vector of integers. Range of keepX values to test (default: seq(10, 100, 10)).
n_bootstrap	Integer. Number of bootstrap samples (default: 100).
threshold	Numeric. Stability threshold between 0 and 1 (default: 0.8). Variables selected in at least this proportion of bootstrap samples are considered stable.
method	Character. Method to use: "sparse_plsda" or "sparse_o2pls".
parallel	Logical. Use parallel processing (default: TRUE).
ncores	Integer. Number of cores for parallel processing (default: NULL, auto-detect).

Details

Stability selection is a method for controlling the expected number of false positive selections in high-dimensional variable selection. It works by applying the sparse method to many bootstrap samples and identifying variables that are consistently selected across different samples and sparsity levels.

The threshold parameter controls the stringency of selection. Higher values (closer to 1) result in more conservative selection with fewer false positives but potentially more false negatives.

Value

stability_selection object containing selection probabilities and stable variables.

Author(s)

Kai Guo

See Also

[sparse_plsda](#), [sparse_o2pls](#), [plot.stability_selection](#)

Examples

```
# Example 1: Stability selection for sparse PLS-DA
set.seed(456)
n <- 100
p <- 150

# Generate data with stable and unstable signals
X <- matrix(rnorm(n * p), n, p)

# Create stable signal in first 15 variables
stable_signal <- matrix(rnorm(n * 2), n, 2)
X[, 1:15] <- stable_signal %*% matrix(rnorm(2 * 15), 2, 15) +
  matrix(rnorm(n * 15, sd = 0.3), n, 15)

# Create unstable/noise signal in variables 16-30
X[, 16:30] <- X[, 16:30] + matrix(rnorm(n * 15, sd = 0.8), n, 15)

# Create classes based on stable signal
class_probs <- plogis(rowSums(X[, 1:15]) - mean(rowSums(X[, 1:15])))
Y <- factor(ifelse(runif(n) < class_probs, "Class1", "Class2"))

colnames(X) <- paste0("Var_", 1:p)

# Perform stability selection (reduced parameters for quick example)
stability_result <- stability_selection(
  X = X,
  Y = Y,
  nc = 2,
  keepX_range = seq(10, 40, 10), # Test fewer sparsity levels
  n_bootstrap = 50,             # Fewer bootstrap samples for speed
```

```

    threshold = 0.7,                # 70% stability threshold
    method = "sparse_plsda",
    parallel = FALSE,               # Disable parallel for example
    ncores = 1
  )

# View results
print(stability_result)

# Extract stable variables for each component
stable_comp1 <- stability_result$stable_variables$Component_1
stable_comp2 <- stability_result$stable_variables$Component_2

if(!is.null(stable_comp1)) {
  cat("Stable variables in Component 1:\n")
  print(stable_comp1)
}

if(!is.null(stable_comp2)) {
  cat("Stable variables in Component 2:\n")
  print(stable_comp2)
}

# Plot stability results
plot(stability_result, type = "heatmap", component = 1)
plot(stability_result, type = "barplot", component = 1, top_n = 30)

# Example 2: Stability selection for sparse O2PLS
# Generate correlated X and Y matrices
Y_matrix <- matrix(rnorm(n * 50), n, 50)
# Add correlation with X
Y_matrix[, 1:10] <- X[, 1:10] + matrix(rnorm(n * 10, sd = 0.5), n, 10)

colnames(Y_matrix) <- paste0("Y_", 1:50)

# Stability selection for O2PLS
## Not run:
stability_o2pls <- stability_selection(
  X = X,
  Y = Y_matrix,
  nc = 2,
  keepX_range = seq(15, 45, 15),
  n_bootstrap = 100,
  threshold = 0.8,
  method = "sparse_o2pls",
  parallel = TRUE,
  ncores = 2
)

# Extract stable variables for both X and Y
stable_X <- stability_o2pls$stable_variables$Component_1$Variable_Name
stable_Y <- stability_o2pls$stable_variables$Component_1$Variable_Name

```

```
## End(Not run)

# Example 3: Different stability thresholds
# Compare conservative vs liberal thresholds
## Not run:
# Conservative selection (80% threshold)
stability_conservative <- stability_selection(
  X, Y, nc = 2, threshold = 0.8, n_bootstrap = 100
)

# Liberal selection (60% threshold)
stability_liberal <- stability_selection(
  X, Y, nc = 2, threshold = 0.6, n_bootstrap = 100
)

# Compare number of stable variables
n_stable_cons <- length(stability_conservative$stable_variables$Component_1$Variable)
n_stable_lib <- length(stability_liberal$stable_variables$Component_1$Variable)

cat("Conservative (80%):", n_stable_cons, "stable variables\n")
cat("Liberal (60%):", n_stable_lib, "stable variables\n")

## End(Not run)
```

summary.O2pls

*Summary of an O2PLS object***Description**

Summary of an O2PLS object

Usage

```
## S3 method for class 'O2pls'
summary(object, ...)
```

Arguments

object	a O2pls object
...	For consistency

Value

Detail of O2PLS results

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(50),10,5)
Y <- matrix(rnorm(50),10,5)
object <- o2pls(X,Y,1,1,1)
summary(object)
```

summary.plsda

Summary of an plsda object

Description

Summary of an plsda object

Usage

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

Arguments

object	a plsda object
...	For consistency

Value

Detail of plsda results

Author(s)

Kai Guo

Examples

```
X <- matrix(rnorm(500),10,50)
Y <- rep(c("a","b"),each=5)
fit <- plsda(X,Y,2)
summary(fit)
```

summary.sparse_plsda	<i>Summary of sparse PLS-DA results</i>
----------------------	---

Description

Summary of sparse PLS-DA results

Usage

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

Arguments

object	A sparse_plsda object to summarize
...	Additional arguments (currently unused)

Value

Summary information printed to console

summary.SparseO2pls	<i>Summary method for sparse O2PLS results</i>
---------------------	--

Description

Summary method for sparse O2PLS results

Usage

```
## S3 method for class 'SparseO2pls'  
summary(object, ...)
```

Arguments

object	A SparseO2pls object to summarize
...	Additional arguments (currently unused)

Value

Summary information about the sparse O2PLS model

tune_sparse_keepX	<i>Cross-validation for Sparse PLS-DA</i>
-------------------	---

Description

Performs cross-validation to tune keepX parameters for sparse PLS-DA

Usage

```
tune_sparse_keepX(  
  X,  
  Y,  
  nc,  
  test.keepX = seq(5, 50, 5),  
  validation = "Mfold",  
  folds = 5,  
  dist = "max.dist",  
  nrepeat = 10  
)
```

Arguments

X	Numeric matrix (samples x variables)
Y	Factor or character vector of class labels
nc	Number of components
test.keepX	Vector of keepX values to test
validation	Validation method ("Mfold" or "loo")
folds	Number of cross-validation folds
dist	Distance metric for classification
nrepeat	Number of repetitions for robust estimation

Value

List with optimal keepX and CV results

tune_sparse_o2pls	<i>Cross-validation for Sparse O2PLS</i>
-------------------	--

Description

Performs cross-validation to evaluate sparse O2PLS models

Usage

```
tune_sparse_o2pls(
  X,
  Y,
  nc_range = 1:3,
  nx_range = 0:2,
  ny_range = 0:2,
  keepX_range = seq(10, 50, 10),
  keepY_range = seq(10, 50, 10),
  validation = "Mfold",
  folds = 5,
  measure = "RMSE"
)
```

Arguments

X	Numeric matrix (samples x variables)
Y	Numeric matrix (samples x variables)
nc_range	Range of joint components to test
nx_range	Range of X-orthogonal components to test
ny_range	Range of Y-orthogonal components to test
keepX_range	Range of keepX values to test
keepY_range	Range of keepY values to test
validation	Validation method
folds	Number of folds
measure	Performance measure

Value

TuneResult object with optimal parameters

TuneResult-class	<i>Class "TuneResult"</i>
------------------	---------------------------

Description

Class "TuneResult"

`vip`

Extract the VIP values from the O2PLS-DA object

Description

Extract the VIP values from the O2PLS-DA object

Usage

```
vip(x)
```

Arguments

`x` the o2plsda object or plsda object

Value

a data frame

Index

* classes

02pls-class, [11](#)
\$, Sparse02pls-method, [3](#)
\$, sparse_plsda-method, [3](#)

apply_sparsity, [4](#)

calculate_mahalanobis_distances, [4](#)
calculate_prediction_probabilities, [5](#)
compare_models, [5](#)

loadings, [6](#)
loadings.o2plsda, [7](#)
loadings.plsda, [7](#)

o2cv, [8](#)
o2cv_sparse, [9](#)
o2pls, [10](#), [34](#)
O2pls-class, [11](#)
oplsda, [12](#)

plot.O2pls, [13](#)
plot.o2plsda, [14](#)
plot.plsda, [16](#)
plot.sparse_plsda, [17](#)
plot.Sparse02pls, [18](#)
plot.stability_selection, [21](#), [41](#)
plot.TuneResult, [22](#)
plsda, [23](#)
predict, sparse_plsda-method, [24](#)
predict, Sparse02pls-method, [25](#)
predict.sparse_plsda, [25](#)
predict.Sparse02pls, [26](#)
print.O2pls, [26](#)
print.plsda, [27](#)
print.Sparse02pls, [27](#)

scad_threshold, [28](#)
scores, [28](#)
scores.O2pls, [29](#)
scores.o2plsda, [29](#)

scores.plsda, [30](#)
selected_var_names, [30](#)
selected_vars, [31](#), [34](#)
selected_vars.default, [31](#)
selected_vars.sparse_plsda, [32](#)
selected_vars.Sparse02pls, [32](#)
sparse_o2pls, [33](#), [37](#), [41](#)
sparse_plsda, [36](#), [41](#)
Sparse02pls-class, [39](#)
sparsity_info, [39](#)
sparsity_info, sparse_plsda-method
 (sparsity_info), [39](#)
sparsity_info, Sparse02pls-method
 (sparsity_info), [39](#)
sparsity_info.sparse_o2pls
 (sparsity_info), [39](#)
sparsity_info.sparse_plsda
 (sparsity_info), [39](#)
sparsity_info.Sparse02pls
 (sparsity_info), [39](#)
stability_selection, [34](#), [37](#), [40](#)
summary.O2pls, [43](#)
summary.plsda, [44](#)
summary.sparse_plsda, [45](#)
summary.Sparse02pls, [45](#)

tune_sparse_keepX, [37](#), [46](#)
tune_sparse_o2pls, [47](#)
TuneResult-class, [47](#)

vip, [48](#)