Package 'guidedPLS'

May 8, 2026

Type Package

Title Supervised Dimensional Reduction by Guided Partial Least Squares

Version 1.1.0

Depends R (>= 3.4.0)

Imports irlba, stats

Suggests fields, geigen, knitr, rmarkdown, testthat

Description

Guided partial least squares (guided-PLS) is the combination of partial least squares by singular value decomposition (PLS-SVD) and guided principal component analysis (guided-PCA). This package provides implementations of PLS-SVD, guided-PLS, and guided-PCA for supervised dimensionality reduction. The guided-PCA function (new in v1.1.0) automatically handles mixed data types (continuous and categorical) in the supervision matrix and provides detailed contribution analysis for interpretability. For the details of the methods, see the reference section of GitHub README.md <https://github.com/rikenbit/guidedPLS>.

License MIT + file LICENSE

URL https://github.com/rikenbit/guidedPLS

VignetteBuilder knitr

NeedsCompilation no

Author Koki Tsuyuzaki [aut, cre]

Maintainer Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Repository CRAN

Date/Publication 2025-08-25 14:00:02 UTC

Contents

`guidedPLS-package` *Supervised Dimensional Reduction by Guided Partial Least Squares*

Description

Guided partial least squares (guided-PLS) is the combination of partial least squares by singular value decomposition (PLS-SVD) and guided principal component analysis (guided-PCA). This package provides implementations of PLS-SVD, guided-PLS, and guided-PCA for supervised dimensionality reduction. The guided-PCA function (new in v1.1.0) automatically handles mixed data types (continuous and categorical) in the supervision matrix and provides detailed contribution analysis for interpretability. For the details of the methods, see the reference section of GitHub README.md <https://github.com/rikenbit/guidedPLS>.

Details

The DESCRIPTION file:

Package:	guidedPLS
Type:	Package
Title:	Supervised Dimensional Reduction by Guided Partial Least Squares
Version:	1.1.0
Authors@R:	c(person("Koki", "Tsuyuzaki", role = c("aut", "cre"), email = "k.t.the-answer@hotmail.co.jp"))
Depends:	R (>= 3.4.0)
Imports:	irlba, stats
Suggests:	fields, geigen, knitr, rmarkdown, testthat
Description:	Guided partial least squares (guided-PLS) is the combination of partial least squares by singular value deco
License:	MIT + file LICENSE
URL:	https://github.com/rikenbit/guidedPLS
VignetteBuilder:	knitr
Author:	Koki Tsuyuzaki [aut, cre]
Maintainer:	Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Index of help topics:

`PLSSVD`	`Partial Least Squares by Singular Value Decomposition (PLS-SVD)`
`dummyMatrix`	`Toy model data for using dNMF, dSVD, dsiNMF, djNMF, dPLS, dNTF, and dNTD`
`guidedPCA`	`Guided PCA (Principal Component Analysis with Label Guidance)`
`guidedPLS`	`Guided Partial Least Squares (guied-PLS)`
`guidedPLS-package`	`Supervised Dimensional Reduction by Guided`

	Partial Least Squares
sPLSDA	Sparse Partial Least Squares Discriminant Analysis (sPLS-DA)
softThr	Soft-thresholding to make a sparse vector sparse
toyModel	Toy model data for using PLSSVD, sPLSDA, and guidedPLS

Author(s)

NA

Maintainer: NA

References

Le Cao, et al. (2008). A Sparse PLS for Variable Selection when Integrating Omics Data. *Statistical Applications in Genetics and Molecular Biology*, 7(1)

Reese S E, et al. (2013). A new statistic for identifying batch effects in high-throughput genomic data that uses guided principal component analysis. *Bioinformatics*, 29(22), 2877-2883

See Also

toyModel,PLSSVD,sPLSDA,guidedPLS

Examples

```
ls("package:guidedPLS")
```

---

dummyMatrix *Toy model data for using dNMF, dSVD, dsiNMF, djNMF, dPLS, dNTF, and dNTD*

Description

A label vector is converted to a dummy matrix.

Usage

```
dummyMatrix(y, center=TRUE)
```

Arguments

y	A label vector to specify the group of data.
center	An option to center the rows of matrix (Default: TRUE).

Value

A matrix is generated. The number of row is equal to the length of y and the number of columns is the number of unique elements of y.

Author(s)

Koki Tsuyuzaki

Examples

```
y <- c(1, 3, 2, 1, 4, 2)
dummyMatrix(y)
```

---

guidedPCA *Guided PCA (Principal Component Analysis with Label Guidance)*

Description

Performs guided PCA by finding principal components that maximize covariance between data matrix X and label/metadata matrix Y. This method extends PLSSVD to automatically handle mixed data types and provide detailed contribution analysis.

Usage

```
guidedPCA(X, Y, k = NULL, center_X = TRUE, scale_X = TRUE,
          normalize_Y = TRUE, contribution = TRUE,
          deflation = FALSE, fullrank = TRUE, verbose = FALSE)
```

Arguments

X	A numeric matrix (samples x features)
Y	A matrix or data.frame with label/metadata (samples x variables). Can contain any mix of numeric (continuous), factor, character (categorical), or logical columns. Each column type is handled appropriately.
k	Number of components to compute (default: min dimensions)
center_X	Logical, whether to center X columns (default: TRUE)
scale_X	Logical, whether to scale X columns to unit variance (default: TRUE)
normalize_Y	Logical, whether to normalize Y columns to unit L2 norm (default: TRUE). This is recommended to balance contributions from different metadata types.
contribution	Logical, whether to calculate feature contributions (default: TRUE)
deflation	Logical, whether to use deflation for sequential component extraction (default: FALSE)
fullrank	Logical, whether to use full SVD or truncated SVD (default: TRUE)
verbose	Logical, whether to print progress messages (default: FALSE)

Details

The algorithm works as follows:

1. Y preprocessing: Mixed data types in Y are handled automatically: - Categorical variables (factor/character) are converted to dummy variables - Continuous variables (numeric) are used as-is - Logical variables are converted to 0/1 - Missing values are handled (NA in factors become a separate category, NA in numerics become 0)

2. Normalization: When normalize_Y=TRUE (default), each Y column is normalized to unit L2 norm. This ensures equal weight across different metadata types, preventing continuous variables with large scales from dominating categorical ones.

3. Core computation: Computes SVD of the cross-product matrix M = X^T Y, where X is the centered/scaled data matrix and Y is the normalized metadata matrix. This finds linear combinations that maximize covariance between X and Y.

Value

A list of class "guidedPCA" containing:

- loadingX: Loading matrix for X (features x components)
- loadingY: Loading matrix for Y (dummy variables x components)
- scoreX: Score matrix for X (samples x components)
- scoreY: Score matrix for Y (samples x components)
- d: Singular values
- Y_dummy: The dummy-encoded Y matrix used internally
- Y_groups: Group labels for dummy variables
- contrib_features: Feature contributions to each component (if contribution=TRUE)
- contrib_groups: Grouped contributions by original Y variables (if contribution=TRUE)
- variance_explained: Variance explained by each component

Author(s)

Koki Tsuyuzaki

References

Reese S E, et al. A new statistic for identifying batch effects in high-throughput genomic data that uses guided principal component analysis. Bioinformatics, 29(22), 2877-2883, 2013

Examples

```
# Example with mixed data types
X <- matrix(rnorm(100*50), 100, 50)
Y <- data.frame(
  celltype = factor(sample(c("A", "B", "C"), 100, replace=TRUE)),
  treatment = factor(sample(c("ctrl", "treated"), 100, replace=TRUE)),
  score = rnorm(100)
)
```

```
result <- guidedPCA(X, Y, k=3)
print(result)
summary(result)
```

---

guidedPLS *Guided Partial Least Squares (guied-PLS)*

Description

Four matrices X1, X2, Y1, and Y2 are required. X1 and Y1 are supposed to share the rows, X2 and Y2 are supposed to share the rows, and Y1 and Y2 are supposed to share the columns.

Usage

```
guidedPLS(X1, X2, Y1, Y2, k=.minDim(X1, X2, Y1, Y2),
    cortest=FALSE, fullrank=TRUE, sumcor=FALSE, lambda=1e-6, verbose=FALSE)
```

Arguments

X1	The input matrix which has N-rows and M-columns.
Y1	The input matrix which has N-rows and L-columns.
X2	The input matrix which has O-rows and P-columns.
Y2	The input matrix which has O-rows and L-columns.
k	The number of low-dimension (k < {N, M, L, O}, Default: .minDim(X1, X2, Y1, Y2))
cortest	If cortest is set as TRUE, t-test of correlation coefficient is performed (Default: FALSE)
fullrank	If fullrank is set as TRUE, irlba is used, otherwise fullrank SVD is used (Default: TRUE)
sumcor	If sumcor is set as TRUE, SUMCOR-based CCA using generalized eigenvalue decomposition is performed instead of SVD-based approach. This maximizes Tr(W1^T X1^T Y1 Y2^T X2 W2) subject to W1^T X1^T X1 W1 = I and W2^T X2^T X2 W2 = I (Default: FALSE). Requires the geigen package.
lambda	Regularization parameter for numerical stability in SUMCOR-based CCA. Only used when sumcor=TRUE (Default: 1e-6). Larger values provide more regularization.
verbose	Verbose option (Default: FALSE)

Value

res: object of svd() loadingYX1: Loading vector to project X1 to lower dimension via Y1 (M times k). loadingYX2: Loading vector to project X2 to lower dimension via Y2 (P times k). scoreX1: Projected X1 (N times k) scoreX2: Projected X2 (O times k) scoreYX1: Projected YX1 (L times k) scoreYX2: Projected YX2 (L times k) corYX1: Correlation Coefficient (Default: NULL) corYX2: Correlation Coefficient (Default: NULL) pvalYX1: P-value vector of corYX1 (Default: NULL)

pvalYX2: P-value vector of corYX2 (Default: NULL) qvalYX1: Q-value vector of BH method against pvalYX1 (Default: NULL) qvalYX2: Q-value vector of BH method against pvalYX2 (Default: NULL)

Author(s)

Koki Tsuyuzaki

References

Le Cao, et al. (2008). A Sparse PLS for Variable Selection when Integrating Omics Data. *Statistical Applications in Genetics and Molecular Biology*, 7(1)

Reese S E, et al. (2013). A new statistic for identifying batch effects in high-throughput genomic data that uses guided principal component analysis. *Bioinformatics*, 29(22), 2877-2883

Examples

```
# Test data
data <- toyModel()

# Simple usage
out <- guidedPLS(X1=data$X1, X2=data$X2, Y1=data$Y1, Y2=data$Y2, k=4)
```

PLSSVD *Partial Least Squares by Singular Value Decomposition (PLS-SVD)*

Description

Two matrices X and Y sharing a row are required

Usage

```
PLSSVD(X, Y, k=.minDim(X, Y), cortest=FALSE,
  deflation=FALSE, fullrank=TRUE, verbose=FALSE)
```

Arguments

X	The input matrix which has N-rows and M-columns.
Y	The input matrix which has N-rows and L-columns.
k	The number of low-dimension (k < {N, M, L}, Default: .minDim(X, Y))
cortest	If cortest is set as TRUE, t-test of correlation coefficient is performed (Default: FALSE)
deflation	If deflation is set as TRUE, the score vectors are made orthogonal, otherwise the loading vectors are made orthogonal (Default: FALSE)
fullrank	If fullrank is set as TRUE, irlba is used, otherwise fullrank SVD is used (Default: TRUE)
verbose	Verbose option (Default: FALSE)

Value

scoreX : Score matrix which has M-rows and K-columns. loadingX : Loading matrix which has N-rows and K-columns. scoreY : Score matrix which has L-rows and K-columns. loadingY : Loading matrix which has N-rows and K-columns. d : K-length singular value vector of the cross-product matrix X'Y. corX: Correlation Coefficient (Default: NULL) corY: Correlation Coefficient (Default: NULL) pvalX: P-value vector of corX (Default: NULL) pvalY: P-value vector of corY (Default: NULL) qvalX: Q-value vector of BH method against pvalX (Default: NULL) qvalY: Q-value vector of BH method against pvalY (Default: NULL)

Author(s)

Koki Tsuyuzaki

References

Le Cao, et al. (2008). A Sparse PLS for Variable Selection when Integrating Omics Data. *Statistical Applications in Genetics and Molecular Biology*, 7(1)

Examples

```
# Test data
data <- toyModel()

# Simple usage
out <- PLSSVD(X=data$X1, Y=data$Y1, k=4)
```

---

softThr *Soft-thresholding to make a sparse vector sparse*

Description

The degree of the sparseness of vector is controlled by the lambda parameter.

Usage

```
softThr(y, lambda=1)
```

Arguments

y	A numerical vector.
lambda	Threshold value to convert a value 0. If the absolute value of an element of vector is less than lambda, the value is converted to 0 (Default: 1).

Value

A numerical vector, whose length is the same as that of y.

Author(s)

Koki Tsuyuzaki

Examples

```
y <- seq(-2, 2, 0.1)
softThr(y)
```

---

sPLSDA *Sparse Partial Least Squares Discriminant Analysis (sPLS-DA)*

Description

Two matrices X and Y sharing a row are required

Usage

```
sPLSDA(X, Y, k=.minDim(X, Y), cortest=FALSE, lambda=1, thr=1e-10, fullrank=TRUE,
    num.iter=10, verbose=FALSE)
```

Arguments

X	The input matrix which has N-rows and M-columns.
Y	The input matrix which has N-rows and L-columns.
k	The number of low-dimension (k < {N, M, L}, Default: .minDim(X, Y))
cortest	If cortest is set as TRUE, t-test of correlation coefficient is performed (Default: FALSE)
lambda	Penalty parameter to control the sparseness of u and v. The larger the value, the sparser the solution (Default: 1).
thr	Threshold to stop the iteration (Default: 1e-10).
fullrank	If fullrank is set as TRUE, irlba is used, otherwise fullrank SVD is used (Default: TRUE)
num.iter	The number of iterations in each rank (Default: 10)
verbose	Verbose option (Default: FALSE)

Value

scoreX : Score matrix which has M-rows and K-columns. loadingX : Loading matrix which has N-rows and K-columns. scoreY : Score matrix which has L-rows and K-columns. loadingY : Loading matrix which has N-rows and K-columns. d : K-length singular value vector of the cross-product matrix X'Y. corX: Correlation Coefficient (Default: NULL) corY: Correlation Coefficient (Default: NULL) pvalX: P-value vector of corX (Default: NULL) pvalY: P-value vector of corY (Default: NULL) qvalX: Q-value vector of BH method against pvalX (Default: NULL) qvalY: Q-value vector of BH method against pvalY (Default: NULL)

Author(s)

Koki Tsuyuzaki

References

Le Cao, et al. (2008). A Sparse PLS for Variable Selection when Integrating Omics Data. *Statistical Applications in Genetics and Molecular Biology*, 7(1)

Examples

```
# Test data
data <- toyModel()

# Simple usage
out <- sPLSDA(X=data$X1, Y=data$Y1, k=4)
```

---

toyModel *Toy model data for using PLSSVD, sPLSDA, and guidedPLS*

Description

The data is used for confirming the algorithm are properly working.

Usage

```
toyModel(model="Easy", seeds=123)
```

Arguments

model	"Easy" and "Hard" are available (Default: "Easy").
seeds	Random number for setting set.seeds in the function (Default: 123).

Value

A list object containing a set of matrices X1, X2, Y1, Y1_dummy, Y2, Y1_dummy.

Author(s)

Koki Tsuyuzaki

See Also

PLSSVD,sPLSDA,guidedPLS

Examples

```
data <- toyModel(seeds=123)
```

Index