Package ‘whitening’

November 7, 2021

Version 1.3.0
Date 2021-11-07
Title Whitening and High-Dimensional Canonical Correlation Analysis
Author Korbinian Strimmer, Takoua Jendoubi, Agnan Kessy, Alex Lewin
Maintainer Korbinian Strimmer <strimmerlab@gmail.com>
Depends R (>= 3.0.2), corpcor (>= 1.6.10)
Imports stats
Suggests
License GPL (>= 3)
URL https://strimmerlab.github.io/software/whitening/
NeedsCompilation no
Repository CRAN
Date/Publication 2021-11-07 18:20:02 UTC

R topics documented:

whitening-package .......................................................... 2
corplot ............................................................................. 2
forina1986 ................................................................. 3
lusc .................................................................................. 5
nutrimouse ................................................................. 6
scca ................................................................................. 8
simOrtho ................................................................. 10
The whitening Package

Description

The "whitening" package implements the whitening methods (ZCA, PCA, Cholesky, ZCA-cor, and PCA-cor) discussed in Kessy, Lewin, and Strimmer (2018) as well as the whitening approach to canonical correlation analysis allowing negative canonical correlations described in Jendoubi and Strimmer (2019).

Author(s)

Korbinian Strimmer (https://strimmerlab.github.io/) with Takoua Jendoubi, Agnan Kessy, and Alex Lewin.

References


Website: https://strimmerlab.github.io/software/whitening/

See Also

whiteningMatrix, whiten, whiteningCrossCov, cca, and scca.

Plots of Correlations and Loadings

corplot computes the correlation within and between X and Y and displays the three corresponding matrices visually.
loadplot computes the squared loadings for X and Y and plots the resulting matrices.

Usage

corplot(cca.out, X, Y)
loadplot(cca.out, numScores)
forina1986

Arguments

cca.out  output from the scca or cca function.
X, Y  input data matrices.
numScores  number of CCA scores shown in plot.

Value

A plot.

Author(s)

Part of the plot code was adapted from the img.matcor function in the CCA package and from the image.plot function in the fields package.

See Also

scca.

forina1986  Forina 1986 Wine Data - Extended UCI Wine Data

Description

The forina1986 dataset describes 27 properties of 178 samples of wine from three grape varieties (59 Barolo, 71 Grignolino, 48 Barbera) as reported in Forina et al. (1986).

Usage

data(forina1986)

Format

A list containing the following components:
  attribs collects measurements for 27 attributes of 178 wine samples.
  type describes the variety ("Barolo", "Grignolino", or "Barbera").

Details


The UCI wine data set (https://archive.ics.uci.edu/ml/datasets/wine) is a subset of the Forina et al. (1986) data set comprising only 13 variables.
Source


References


Examples

# load whitening library
library("whitening")

# load Forina 1986 wine data set
data(forina1986)
table(forina1986$type)
# Barolo Grignolino Barbera
# 59 71 48
dim(forina1986$attrib)
# 178 27
colnames(forina1986$attrib)
# [1] "Alcohol" "Sugar-free extract"
# [3] "Fixed acidity" "Tartaric acid"
# [5] "Malic acid" "Uronic acids"
# [7] "pH" "Ash"
# [9] "Alkalinity of ash" "Potassium"
#[11] "Calcium" "Magnesium"
#[13] "Phosphate" "Chloride"
#[15] "Total phenols" "Flavanoids"
#[17] "Nonflavanoid phenols" "Proanthocyanins"
#[19] "Color intensity" "Hue"
#[21] "OD280/OD315 of diluted wines" "OD280/OD315 of flavonoids"
#[23] "Glycerol" "2-3-butanediol"
#[25] "Total nitrogen" "Proline"
#[27] "Methanol"

# PCA-cor whitened data
Z = whiten(forina1986$attrib, method="PCA-cor")
wt = as.integer(forina1986$type)
plot(Z[,1], Z[,2], xlab=expression(paste(Z[1])), ylab=expression(paste(Z[2])),
main="Forina 1986 Wine Data", sub="PCA-cor Whitening", col=wt, pch=wt+14)
legend("topright", levels(forina1986$type)[1:3], col=1:3, pch=(1:3)+14 )
## relationship to UCI wine data

UCI wine data is a subset

```r
```

# two small differences compared to UCI wine data matrix

```r
uciwine.attrib[172,"Color intensity"] # 9.9 but 9.899999 in UCI matrix
uciwine.attrib[71,"Hue"] # 0.91 but 0.906 in UCI matrix
```

---

### Description

A preprocessed sample of gene expression and methylation data as well as selected clinical co-
variates for 130 patients with lung squamous cell carcinoma (LUSC) as available from The Cancer
Genome Atlas (TCGA) database (Kandoth et al. 2013).

### Usage

```r
data(lusc)
```

### Format

- `lusc$rnaseq2` is a 130 x 206 matrix containing the calibrated gene expression levels of 206 genes for 130 patients.
- `lusc$methyl` is a 130 x 234 matrix containing the methylation levels of 234 probes for 130 patients.
- `sex` is a vector recording the sex (male vs. female) of the 130 patients.
- `packs` is the number of cigarette packs per year smoked by each patient.
- `survivalTime` is number of days to last follow-up or the days to death.
- `censoringStatus` is the vital status (0=alive, 1=dead).

### Details

This data set is used to illustrate CCA-based data integration in Jendoubi and Strimmer (2019) and also described in Wan et al. (2016).

### Source

The data were retrieved from TCGA (Kandoth et al. 2014) using the TCGA2STAT tool following the
guidelines and the preprocessing steps detailed in Wan et al. (2016).
References


Examples

```r
# load whitening library
library("whitening")

# load TGCA LUSC data set
data(lusc)

names(lusc)  # "rnaseq2" "methyl" "sex" "packs"
# "survivalTime" "censoringStatus"

dim(lusc$rnaseq2)  # 130 206 gene expression

dim(lusc$methyl)  # 130 234 methylation level

## Not run:
library("survival")
s = Surv(lusc$survivalTime, lusc$censoringStatus)
plot(survfit(s ~ lusc$sex), xlab = "Years", ylab = "Probability of survival", lty=c(2,1), lwd=2)
legend("topright", legend = c("male", "female"), lty =c(1,2), lwd=2)

## End(Not run)
```

Nutrimouse Data

Description

The nutrimouse dataset is a collection of gene expression and lipid measurements collected in a nutrigenomic study in the mouse studying 40 animals by Martin et al. (2007).

Usage

data(nutrimouse)
**nutrimouse**

**Format**

A list containing the following components:

gene collects gene expression of 120 genes in liver tissue for 40 mice.

lipid collects concentrations of 21 lipids for 40 mice.

diet describes the diet of each mouse ("coc", "fish", "lin", "ref", or "sun").

genotype describes the genotype of each mouse: wild type ("wt") or PPARalpha deficient ("ppar").

**Details**

This data set is used to illustrate CCA-based data integration in Jendoubi and Strimmer (2019) and is also described in Gonzalez et al. (2008).

**Source**

The original data are available in the CCA package by Gonzalez et al. (2008), see their function nutrimouse.

**References**


**Examples**

```r
# load whitening library
library("whitening")

# load nutrimouse data set
data(nutrimouse)

dim(nutrimouse$gene) # 40 120
dim(nutrimouse$lipid) # 40 21
levels(nutrimouse$diet ) # "coc" "fish" "lin" "ref" "sun"
levels(nutrimouse$genotype ) # "wt" "ppar"
```
Perform Canonical Correlation Analysis

Description

scca computes canonical correlations and directions using a shrinkage estimate of the joint correlation matrix of $X$ and $Y$.

cca computes canonical correlations and directions based on empirical correlations.

Usage

scca(X, Y, lambda.cor, scale=TRUE, verbose=TRUE)
cca(X, Y, scale=TRUE)

Arguments

- **X**: First data matrix, with samples in rows and variables in columns.
- **Y**: Second data matrix, with samples in rows and variables in columns.
- **lambda.cor**: Shrinkage intensity for estimating the joint correlation matrix - see cor.shrink. If not specified this will be estimated from the data.
- **scale**: Determines whether canonical directions are computed for standardized or raw data. Note that if data are not standardized the canonical directions contain the scale of the variables.
- **verbose**: Report shrinkage intensities.

Details

The canonical directions in this function are scaled in such a way that they correspond to whitening matrices - see Jendoubi and Strimmer (2019) for details. Note that the sign convention for the canonical directions employed here allows purposely for both positive and negative canonical correlations.

The function scca uses some clever matrix algebra to avoid computation of full correlation matrices, and hence can be applied to high-dimensional data sets - see Jendoubi and Strimmer (2019) for details.

cca it is a shortcut for running scca with lambda.cor=0 and verbose=FALSE.

If scale=FALSE the standard deviations needed for the canonical directions are estimated by apply(X, 2, sd) and apply(Y, 2, sd).

If $X$ or $Y$ contains only a single variable the correlation-adjusted cross-correlations $K$ reduce to the CAR score (see function carscore in the care package) described in Strimmer and Zuber (2011).
Value

scca and cca return a list with the following components:

- **K** - the correlation-adjusted cross-correlations.
- **lambda** - the canonical correlations.
- **WX** and **WY** - the whitening matrices for X and Y, with canonical directions in the rows. If scale=FALSE then canonical directions include scale of the data, if scale=TRUE then only correlations are needed to compute the canonical directions.
- **PhiX** and **PhiY** - the loadings for X and Y. If scale=TRUE then these are the correlation loadings, i.e. the correlations between the whitened variables and the original variables.
- **scale** - whether data was standardized (if scale=FALSE then canonical directions include scale of the data).
- **lambda.cor** - shrinkage intensity used for estimating the correlations (0 for empirical estimator)
- **lambda.cor.estimated** - indicates whether shrinkage intensity was specified or estimated.

Author(s)

Korbinian Strimmer (https://strimmerlab.github.io) with Takoua Jendoubi.

References


See Also

cancor and whiteningMatrix.

Examples

# load whitening library
library("whitening")

# example data set
data(LifeCycleSavings)
X = as.matrix( LifeCycleSavings[, 2:3] )
Y = as.matrix( LifeCycleSavings[, -(2:3)] )
n = nrow(X)
colnames(X) # "pop15" "pop75"
colnames(Y) # "sr" "dpi" "ddpi"

# CCA

cca.out = cca(X, Y, scale=TRUE)
cca.out$lambda # canonical correlations
cca.out$WX # whitening matrix / canonical directions X
simOrtho

Simulate Random Orthogonal Matrix

Description

simOrtho generates a random orthogonal matrix.

Usage

simOrtho(d)

Arguments

d

The dimension of the orthogonal matrix.

Details

The algorithm is based on QR decomposition of a random matrix, see Section 3 page 404 in Stewart (1980).
whiten

Value
simOrtho returns a real matrix of size $d \times d$.

Author(s)

References

See Also
whiteningMatrix

Examples
# load whitening library
library("whitening")

# simulate random orthogonal matrix
Q = simOrtho(4) # matrix of dimension 4x4
Q
zapsmall( crossprod(Q) )
zapsmall( tcrossprod(Q) )

whiten
Whiten Data Matrix

Description
whiten whitens a data matrix $X$ using the empirical covariance matrix $\text{cov}(X)$ as basis for computing the whitening transformation.

Usage

Arguments
X Data matrix, with samples in rows and variables in columns.
center Center columns to mean zero.
method Determines the type of whitening transformation.
The following six different whitening approaches can be selected:

method="ZCA": ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and original variables is maximal.

method="ZCA-cor": Likewise, ZCA-cor whitening leads to whitened variables that are maximally correlated (on average) with the original variables.

method="PCA": In contrast, PCA whitening lead to maximally compressed whitened variables, as measured by squared covariance.

method="PCA-cor": PCA-cor whitening is similar to PCA whitening but uses squared correlations.

method="Chol-prec" and method="Cholesky": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields an upper triangular positive diagonal whitening matrix and lower triangular positive diagonal cross-covariance and cross-correlation matrices.

method="Chol-cov": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields a lower triangular positive diagonal whitening matrix and upper triangular positive diagonal cross-covariance and cross-correlation matrices.

ZCA-cor whitening is implicitly employed in computing CAT and CAR scores used for variable selection in classification and regression, see the functions catscore in the sda package and carscore in the care package.

In both PCA and PCA-cor whitening there is a sign-ambiguity in the eigenvector matrices. In order to resolve the sign-ambiguity we use eigenvector matrices with a positive diagonal. This has the effect to make cross-correlations and cross-correlations positive diagonal for PCA and PCA-cor.

For details see Kessy, Lewin, and Strimmer (2018).

whiten returns the whitened data matrix $Z = XW'$.

Korbinian Strimmer (https://strimmerlab.github.io) with Agnan Kessy and Alex Lewin.


whiteningMatrix.

# load whitening library
library("whitening")

#######
# example data set
# Bull. Am. Iris Soc. 59: 2--5
data(“iris”)
X = as.matrix(iris[,1:4])
d = ncol(X) # 4
n = nrow(X) # 150
colnames(X) # "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"

# whitened data
Z.ZCAcor = whiten(X, method="ZCA-cor")

# check covariance matrix
zapsmall( cov(Z.ZCAcor) )

whiteningCrossCov

Compute Whitening Cross-Covariance Matrix

Description

whiteningCrossCov computes the cross-covariance $\Phi = \text{Cov}(z, x)$ between the whitened and the original variables.

Usage

whiteningCrossCov(Sigma, method=c("ZCA", "ZCA-cor", "PCA", "PCA-cor", "Chol-prec", "Chol-cov", "Cholesky"))

Arguments

Sigma Covariance matrix.

method Determines the type of whitening transformation.

Details

$\Phi = \text{Cov}(z, x)$ is the cross-covariance matrix between the whitened and the original variables. It satisfies $\Phi'\Phi = \text{Sigma}$. The transpose of the cross-covariance matrix is the inverse of the whitening matrix so that $\Phi' = W^{-1}$. The cross-covariance matrix is therefore relevant in inverse whitening transformations (=coloring transformations) $x = \Phi'z$.

The following six different whitening approaches can be selected:

- method="ZCA": ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and original variables is maximal.
- method="ZCA-cor": Likewise, ZCA-cor whitening leads to whitened variables that are maximally correlated (on average) with the original variables.
method="PCA": In contrast, PCA whitening lead to maximally compressed whitened variables, as measured by squared covariance.

method="PCA-cor": PCA-cor whitening is similar to PCA whitening but uses squared correlations.

method="Chol-prec" and method="cholesky": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields an upper triangular positive diagonal whitening matrix and lower triangular positive diagonal cross-covariance and cross-correlation matrices.

method="Chol-cov": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields a lower triangular positive diagonal whitening matrix and upper triangular positive diagonal cross-covariance and cross-correlation matrices.

ZCA-cor whitening is implicitly employed in computing CAT and CAR scores used for variable selection in classification and regression, see the functions catscore in the sda package and carscore in the care package.

In both PCA and PCA-cor whitening there is a sign-ambiguity in the eigenvector matrices. In order to resolve the sign-ambiguity we use eigenvector matrices with a positive diagonal. This has the effect to make cross-correlations and cross-correlations positive diagonal for PCA and PCA-cor.

For details see Kessy, Lewin, and Strimmer (2018).

Value

whiteningCrossCov returns a square cross-covariance matrix $\Phi$.

Author(s)


References


See Also

whiteningMatrix.

Examples

# load whitening library
library("whitening")

###

# example data set
# Bull. Am. Iris Soc. 59: 2--5
data("iris")
X = as.matrix(iris[,1:4])
d = ncol(X) # 4
n = nrow(X) # 150
colnames(X) # "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"
# estimate covariance
S = cov(X)

# ZCA-cor whitening matrix
W.ZCAcor = whiteningMatrix(S, method="ZCA-cor")

# ZCA-cor cross-covariance matrix
Phi.ZCAcor = whiteningCrossCov(S, method="ZCA-cor")

# check constraint of cross-covariance matrix
crossprod(Phi.ZCAcor)
S

# transpose of cross-covariance matrix is equal to the inverse whitening matrix
t(Phi.ZCAcor)
solve(W.ZCAcor)

whiteningMatrix

Compute Whitening Matrix

Description
whiteningMatrix computes the whitening matrix W.

Usage
whiteningMatrix(Sigma, method=c("ZCA", "ZCA-cor", "PCA", "PCA-cor",
"Chol-prec", "Chol-cov", "Cholesky"))

Arguments
Sigma Covariance matrix.
method Determines the type of whitening transformation.

Details
Whitening is a linear transformation $z = Wx$ where the whitening matrix satisfies the constraint $W'W = Sigma^{-1}$ where $Sigma = Cov(x)$.

This function implements various natural whitening transformations discussed in Kessy, Lewin, and Strimmer (2018).

The following six different whitening approaches can be selected:

method="ZCA": ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and original variables is maximal.

method="ZCA-cor": Likewise, ZCA-cor whitening leads to whitened variables that are maximally correlated (on average) with the original variables.
method="PCA": In contrast, PCA whitening lead to maximally compressed whitened variables, as measured by squared covariance.

method="PCA-cor": PCA-cor whitening is similar to PCA whitening but uses squared correlations.

method="Chol-prec" and method="Cholesky": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields an upper triangular positive diagonal whitening matrix and lower triangular positive diagonal cross-covariance and cross-correlation matrices.

method="Chol-cov": computes a whitening matrix by applying Cholesky decomposition on the precision matrix. This yields a lower triangular positive diagonal whitening matrix and upper triangular positive diagonal cross-covariance and cross-correlation matrices.

ZCA-cor whitening is implicitly employed in computing CAT and CAR scores used for variable selection in classification and regression, see the functions catscore in the sda package and carscore in the care package.

In both PCA and PCA-cor whitening there is a sign-ambiguity in the eigenvector matrices. In order to resolve the sign-ambiguity we use eigenvector matrices with a positive diagonal. This has the effect to make cross-correlations and cross-correlations positive diagonal for PCA and PCA-cor.

For details see Kessy, Lewin, and Strimmer (2018). Canonical correlation analysis (CCA) can also be understood as a special form form of whitening.

Value

whiteningMatrix returns a square whitening matrix $W$.

Author(s)

Korbinian Strimmer (https://strimmerlab.github.io) with Agnan Kessy and Alex Lewin.

References


See Also

whiteningCrossCov, scca.

Examples

# load whitening library
library("whitening")

# example data set
# Bull. Am. Iris Soc. 59: 2--5
data("iris")
X = as.matrix(iris[,1:4])
d = ncol(X) # 4
```r
n = nrow(X) # 150
colnames(X) # "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"

# estimate covariance
S = cov(X)

# ZCA-cor whitening matrix
W.ZCAcor = whiteningMatrix(S, method="ZCA-cor")

# check of constraint on whitening matrix
crossprod(W.ZCAcor)
solve(S)
```
Index

* datasets
  forina1986, 3
  lusc, 5
  nutrimouse, 6
* multivariate
  scca, 8
  simOrtho, 10
  whiten, 11
  whitening-package, 2
  whiteningCrossCov, 13
  whiteningMatrix, 15
* plot
  corplot, 2
  cancor, 9
  cca, 2, 3
  cca (scca), 8
  cor.shrink, 8
  corplot, 2
  forina1986, 3
  loadplot (corplot), 2
  lusc, 5
  nutrimouse, 6
  scca, 2, 3, 8, 16
  simOrtho, 10
  whiten, 2, 11
  whitening-package, 2
  whiteningCrossCov, 2, 13, 16
  whiteningMatrix, 2, 9, 11, 12, 14, 15