Package ‘whitening’

October 12, 2022

Version 1.4.0

Date 2022-06-07

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 (>= 3.5.0), corpcor (>= 1.6.10)

Imports stats

Suggests

Description Implements the whitening methods (ZCA, PCA, Cholesky, ZCA-cor, and PCA-cor) discussed in Kessy, Lewin, and Strimmer (2018) ``Optimal whitening and decorrelation'', <doi:10.1080/00031305.2016.1277159>, as well as the whitening approach to canonical correlation analysis allowing negative canonical correlations described in Jendoubi and Strimmer (2019) ``A whitening approach to probabilistic canonical correlation analysis for omics data integration'', <doi:10.1186/s12859-018-2572-9>. The package also offers functions to simulate random orthogonal matrices, compute (correlation) loadings and explained variation. It also contains four example data sets (extended UCI wine data, TCGA LUSC data, nutrimouse data, extended pitprops data).

License GPL (>= 3)

URL `https://strimmerlab.github.io/software/whitening/`

NeedsCompilation no

Repository CRAN

Date/Publication 2022-06-07 12:20:02 UTC

R topics documented:

`whitening-package` *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

Kessy, A., A. Lewin, and K. Strimmer. 2018. Optimal whitening and decorrelation. The American Statistician. 72: 309-314. <DOI:10.1080/00031305.2016.1277159>

Jendoubi, T., and K. Strimmer 2019. A whitening approach to probabilistic canonical correlation analysis for omics data integration. BMC Bioinformatics 20: 15. <DOI:10.1186/s12859-018-2572-9>

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

See Also

`whiteningMatrix`, `whiten`, `whiteningLoadings`, `explainedVariation`, `cca`, and `scca`.

`corplot` *Plots of Correlations and Loadings*

Description

`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)
```

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)

Korbinian Strimmer (`https://strimmerlab.github.io`).

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`.

`explainedVariation` *Compute Explained Variation from Loadings*

Description

`explainedVariation` computes the explained variation for each whitened variables from the loadings (both covariance loadings and correlation loadings).

Usage

```
explainedVariation(Phi)
```

Arguments

Phi	Loading matrix (with columns referring to whitened variables).

Details

`explainedVariation` computes for each column of the loading matrix the sum of squares of the elements in that column.

Value

`explainedVariation` returns a vector with the explained variation contributed by each whitened variable.

Author(s)

Korbinian Strimmer (`https://strimmerlab.github.io`).

References

Kessy, A., A. Lewin, and K. Strimmer. 2018. Optimal whitening and decorrelation. The American Statistician. 72: 309-314. <DOI:10.1080/00031305.2016.1277159>

See Also

`whiteningLoadings`

Examples

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

######

# example data set
# E. Anderson. 1935.  The irises of the Gaspe Peninsula.
# 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)

# PCA-cor loadings
ldgs = whiteningLoadings(S, method="PCA-cor")

# Explained variation from correlation loadings
explainedVariation( ldgs$Psi )
```

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

This data set contains the full set of covariates described in Forina et al. (1986) except for Sulphate (variable 15 in Forina et al. 1986). These are: 1) Alcohol, 2) Sugar-free extract, 3) Fixed acidity, 4) Tartaric acid, 5) Malic acid, 6) Uronic acids, 7) pH, 8) Ash, 9) Alcalinity of ash, 10) Potassium, 11) Calcium, 12) Magnesium, 13) Phosphate, 14) Chloride, 15) Total phenols, 16) Flavanoids, 17) Nonflavanoid phenols, 18) Proanthocyanins, 19) Color intensity, 20) Hue, 21) OD280/OD315 of diluted wines, 22) OD280/OD315 of flavonoids, 23) Glycerol, 24) 2-3-butanediol, 25) Total nitrogen, 26) Proline, and 27) Methanol.

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

The original data matrix is available from `https://www.researchgate.net/publication/271908647_Wines_MForina_CArmanino_MCastino_MUbigli_Multivariate_data_analysis_as_discriminating_method_of_the_origin_of_wines_Vitis_25_189-201_1986`.

References

Forina, M., Armanino, C., Castino, M., and Ubigli, M. Multivariate data analysis as a discriminating method of the origin of wines. Vitis 25:189-201 (1986). `https://ojs.openagrar.de/index.php/VITIS/article/view/5950`.

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
uciwine.attrib = forina1986$attrib[, c("Alcohol", "Malic acid", "Ash",
  "Alcalinity of ash", "Magnesium", "Total phenols", "Flavanoids",
  "Nonflavanoid phenols", "Proanthocyanins", "Color intensity", "Hue",
  "OD280/OD315 of diluted wines", "Proline")]

# two small differences compared to UCI wine data matrix
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
```

`lusc` *TCGA LUSC Data*

Description

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

Usage

```
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

Jendoubi, T., Strimmer, K.: A whitening approach to probabilistic canonical correlation analysis for omics data integration. BMC Bioinformatics 20:15 <DOI:10.1186/s12859-018-2572-9>

Kandoth, C., McLellan, M.D., Vandin, F., Ye, K., Niu, B., Lu, C., Xie, M., andJ. F. McMichael, Q.Z., Wyczalkowski, M.A., Leiserson, M.D.M., Miller, C.A., Welch, J.S., Walter, M.J., Wendl, M.C., Ley, T.J., Wilson, R.K., Raphael, B.J., Ding, L.: Mutational landscape and significance across 12 major cancer types. Nature 502, 333–339 (2013). <DOI:10.1038/nature12634>

Wan, Y.-W., Allen, G.I., Liu, Z.: TCGA2STAT: simple TCGA data access for integrated statistical analysis in R. Bioinformatics 32, 952–954 (2016). <DOI:10.1093/bioinformatics/btv677>

Examples

```
# 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 *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)
```

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

Gonzalez, I., Dejean, S., Martin, P.G.P, Baccini, A. CCA: an R package to extend canonical correlation analysis. J. Statist. Software 23:1–13 (2008)

Jendoubi, T., Strimmer, K.: A whitening approach to probabilistic canonical correlation analysis for omics data integration. BMC Bioinformatics 20:15 <DOI:10.1186/s12859-018-2572-9>

Martin, P.G.P., Guillou, H., Lasserre, F., Dejean, S., Lan, A., Pascussi, J.-M., Cristobal, M.S., Legrand, P., Besse, P., Pineau, T.: Novel aspects of PPARalpha-mediated regulation of lipid and xenobiotic metabolism revealed through a multigenomic study. Hepatology 54, 767–777 (2007) <DOI:10.1002/hep.21510>

Examples

```
# 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"
```

`pitprops14` *Pitprops Correlation Data for 14 Variables*

Description

Pit prop timber is used in construction to build mining tunnels. The `pitprops14` data is described in Jeffers (1967) and is a correlation matrix that was calculated from measuring 14 physical properties of 180 pit props made from wood from Corsican pines grown in East Anglia, UK.

Usage

```
data(pitprops14)
```

Format

A correlation matrix of dimension 14 times 14.

Details

The 14 variables are described in Jeffers (1967) as follows:

`topdiam`: the top diameter of the prop in inches; `length`: the length of the prop in inches; `moist`: the moisture content of the prop, expressed as a percentage of the dry weight; `testsg`: the specific gravity of the timber at the time of the test; `ovensg`: the oven-dry specific gravity of the timber; `rinotop`: the number of annual rings at the top of the prop; `ringbut`: the number of annual rings at the base of the prop; `bowmax`: the maximum bow in inches; `bowdist`: the distance of the point of maximum bow from the top of the prop in inches; `whorls`: the number of knot whorls; `clear`: the length of clear prop from the top of the prop in inches; `knots`: the average number of knots per whorl; `diaknot`: the average diameter of the knots in inches; `maxcs`: the maximum compressive strength in lb per square inch.

Source

The data set is printed in Jeffers (1967) in Table 2 and Table 5.

References

Jeffers, J. N. R. 1967. Two case studies in the application of principal component analysis. JRSS C (Applied Statistics) 16: 225-236. <DOI:10.2307/2985919>

Examples

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

# load pitprops14 data set
data(pitprops14)
colnames(pitprops14)

# correlation matrix for the first 13 variables
pitprops13 = pitprops14[1:13, 1:13]

# correlation loadings for PCA whitening
Psi = whiteningLoadings(pitprops13, "PCA")$Psi

# corresponding explained variation
Psi.explained = explainedVariation(Psi)

# the first six whitened variables account for 87% of the variation
cumsum(Psi.explained)/13*100
```

`scca` *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(X, 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 intenstiy was specified or estimated.

Author(s)

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

References

Jendoubi, T., and K. Strimmer 2019. A whitening approach to probabilistic canonical correlation analysis for omics data integration. BMC Bioinformatics 20: 15. <DOI:10.1186/s12859-018-2572-9>

Zuber, V., and K. Strimmer. 2011. High-dimensional regression and variable selection using CAR scores. Statist. Appl. Genet. Mol. Biol. 10: 34. <DOI:10.2202/1544-6115.1730>

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
cca.out$WY      # whitening matrix / canonical directions Y
cca.out$K       # correlation-adjusted cross-correlations
cca.out$PhiX    # correlation loadings X
cca.out$PhiX    # correlation loadings Y

corplot(cca.out, X, Y)
loadplot(cca.out, 2)
# column sums of squared correlation loadings add to 1
colSums(cca.out$PhiX^2)
```

```
# CCA whitened data
CCAX = tcrossprod( scale(X), cca.out$WX )
CCAY = tcrossprod( scale(Y), cca.out$WY )
zapsmall(cov(CCAX))
zapsmall(cov(CCAY))
zapsmall(cov(CCAX,CCAY)) # canonical correlations


# compare with built-in function cancor
# note different signs in correlations and directions!
cancor.out = cancor(scale(X), scale(Y))
cancor.out$cor                  # canonical correlations
t(cancor.out$xcoef)*sqrt(n-1)   # canonical directions X
t(cancor.out$ycoef)*sqrt(n-1)   # canonical directions Y


## see "User guides, package vignettes and other documentation"
## for examples with high-dimensional data using the scca function
```

---

simOrtho *Simulate Random Orthogonal Matrix*

Description

simOrtho generates a random orthogonal matrix.

Usage

```
simOrtho(d, nonNegDiag = FALSE)
```

Arguments

d	The dimension of the orthogonal matrix.
nonNegDiag	force the elements on the diagonal to be nonnegative (default: FALSE).

Details

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

Value

simOrtho returns a real matrix of size d times d.

Author(s)

Korbinian Strimmer (https://strimmerlab.github.io).

References

G.W. Stewart. 1980. The efficient generation of random orthogonal matrices with an application to condition estimators. SIAM J. Numer. Anal. 17:403-409. <DOI:10.1137/0717034>

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 $cov(X)$ as basis for computing the whitening transformation.

Usage

```
whiten(X, center=FALSE, method=c("ZCA", "ZCA-cor", "PCA", "PCA-cor", "Cholesky"))
```

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 (see Details).

Details

The following whitening approaches can be selected:

`method="ZCA"` and `method="ZCA-cov"`: ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and orginal 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" and method="PCA-cov": 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="Cholesky": computes a whitening matrix by applying Cholesky decomposition. This yields both a lower triangular positive diagonal whitening matrix and lower triangular positive diagonal loadings (cross-covariance and cross-correlation).

Note that Cholesky whitening depends on the ordering of input variables. In the convention used here the first input variable is linked with the first latent variable only, the second input variable is linked to the first and second latent variable only, and so on, and the last variable is linked to all latent variables.

ZCA-cor whitening is implicitely 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 so that PCA and PCA-cor cross-correlations and cross-covariances have a positive diagonal for the given ordering of the original variables.

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

Canonical correlation analysis (CCA) can also be understood as a special form of whitening (also implemented in this package).

Value

whiten returns the whitened data matrix $Z = XW^T$.

Author(s)

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

References

Kessy, A., A. Lewin, and K. Strimmer. 2018. Optimal whitening and decorrelation. The American Statistician. 72: 309-314. <DOI:10.1080/00031305.2016.1277159>

See Also

whiteningMatrix, whiteningLoadings, scca.

Examples

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

######

# example data set
# E. Anderson. 1935.  The irises of the Gaspe Peninsula.
# 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) )
```

`whiteningLoadings` *Compute Whitening Loadings*

Description

`whiteningLoading` computes the loadings (= cross-covariance matrix $\Phi = Cov(x, z)$) between the original and the whitened variables as well as the correlation loadings (= cross-correlation matrix $\Psi = Cor(x, z)$). The original variables are in the rows and the whitened variables in the columns.

Usage

```
whiteningLoadings(Sigma, method=c("ZCA", "ZCA-cor", "PCA", "PCA-cor", "Cholesky"))
```

Arguments

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

Details

$\Phi = Cov(x, z)$ is the cross-covariance matrix between the original and the whitened variables. It satisfies $\Phi\Phi^T = \Sigma = Var(x)$. This 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) $x = \Phi z$.

$\Psi = Cor(x, z)$ is the cross-correlation matrix between the original and the whitened variables.

The following different whitening approaches can be selected:

`method="ZCA"`: ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and orginal 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="Cholesky"`: computes a whitening matrix by applying Cholesky decomposition. This yields both a lower triangular positive diagonal whitening matrix and lower triangular positive diagonal loadings (cross-covariance and cross-correlation).

Note that Cholesky whitening depends on the ordering of input variables. In the convention used here the first input variable is linked with the first latent variable only, the second input variable is linked to the first and second latent variable only, and so on, and the last variable is linked to all latent variables.

ZCA-cor whitening is implicitely 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 so that PCA and PCA-cor cross-correlations and cross-covariances have a positive diagonal for the given ordering of the original variables.

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

Value

`whiteningLoadings` returns a list with the following items:

`Phi` - cross-covariance matrix Φ - the loadings.

`Psi` - cross-correlation matrix Ψ - the correlation loadings.

Author(s)

Korbinian Strimmer (`https://strimmerlab.github.io`).

References

Kessy, A., A. Lewin, and K. Strimmer. 2018. Optimal whitening and decorrelation. The American Statistician. 72: 309-314. <DOI:10.1080/00031305.2016.1277159>

See Also

`explainedVariation`, `whiten`, `whiteningMatrix`.

Examples

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

######

# example data set
# E. Anderson. 1935.  The irises of the Gaspe Peninsula.
# 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 loadings
ldgs = whiteningLoadings(S, method="ZCA-cor")
ldgs

# cross-covariance matrix
Phi.ZCAcor = ldgs$Phi

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

# cross-covariance matrix aka loadings is equal to the inverse whitening matrix
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", "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 $W^T W = \Sigma^{-1}$ where $\Sigma = Cov(x)$.

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

The following different whitening approaches can be selected:

`method="ZCA"`: ZCA whitening, also known as Mahalanobis whitening, ensures that the average covariance between whitened and orginal 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="Cholesky": computes a whitening matrix by applying Cholesky decomposition. This yields both a lower triangular positive diagonal whitening matrix and lower triangular positive diagonal loadings (cross-covariance and cross-correlation).

Note that Cholesky whitening depends on the ordering of input variables. In the convention used here the first input variable is linked with the first latent variable only, the second input variable is linked to the first and second latent variable only, and so on, and the last variable is linked to all latent variables.

ZCA-cor whitening is implicitely 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 so that PCA and PCA-cor cross-correlations and cross-covariances have a positive diagonal for the given ordering of the original variables.

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

Canonical correlation analysis (CCA) can also be understood as a special form of whitening (also implemented in this package).

Value

whiteningMatrix returns a square whitening matrix W.

Author(s)

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

References

Kessy, A., A. Lewin, and K. Strimmer. 2018. Optimal whitening and decorrelation. The American Statistician. 72: 309-314. <DOI:10.1080/00031305.2016.1277159>

See Also

whiten, whiteningLoadings, scca.

Examples

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

# example data set
# E. Anderson. 1935.  The irises of the Gaspe Peninsula.
# 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")

# check constraint on the whitening matrix
crossprod(W.ZCAcor)
solve(S)

# whitened data
Z = tcrossprod(X, W.ZCAcor)
Z
zapsmall( cov(Z) )
```

Index