Package ‘ICS’

March 3, 2018

Type Package

Title Tools for Exploring Multivariate Data via ICS/ICA

Version 1.3-1

Date 2018-03-02

Author Klaus Nordhausen, Hannu Oja, David E. Tyler

Maintainer Klaus Nordhausen <klaus.nordhausen@tuwien.ac.at>

Depends R (>= 2.5.0), methods, mvtnorm

Imports survey, graphics

Suggests pixmap, robustbase, MASS, ICSNP

Description Implementation of Tyler, Critchley, Duembgen and Oja's (JRSS B, 2009, <doi:10.1111/j.1467-9868.2009.00706.x>) and Oja, Sirkia and Eriksson's (AJS, 2006, <http://www.ajs.or.at/index.php/ajs/article/view/vol35,%20no2%263%20-%207>) method of two different scatter matrices to obtain an invariant coordinate system or independent components, depending on the underlying assumptions.

LazyLoad yes

Encoding latin1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-03-03 00:53:56 UTC

R topics documented:

`ICS-package` *Tools for Exploring Multivariate Data via ICS/ICA*

Description

Implements the two scatter matrices transformation to obtain an invariant coordinate sytem or independent components, depending on the underlying assumptions. The result of the transformation is an object of the S4 class `ics` which is provided by this package. Besides generic functions to create and work with an `ics` object the package contains also some scatter matrices.

Details

Package:	ICS
Type:	Package
Version:	1.3-1
Date:	2018-03-02
License:	GPL (>= 2)

Some multivariate tests and estimates are not affine equivariant by nature. A possible remedy for the lack of that property is to transform the data points to an invariant coordinate system, construct tests and estimates from the transformed data, and if needed, retransform the estimates back. The use of two different scatter matrices to obtain invariant coordinates is implemeted in this package by the function `ics`. For an invariant coordinate selection no assumptions are made about the data or the scatter matrices and it can be seen as a data transformation method. If the data come, however, from a so called independent component model the `ics` function can recover the independent components and estimate the mixing matrix under general assumptions. The function `ics2` is an augmented version of `ics` which takes also two location vectors into consideration to obtain natural centers and skewness measures for the invariance coordinates. Besides the functions `ics` and `ics2`, which create S4 object of classes `ics` and `ics2`, provides this package tools to work with objects of these classes and some scatter matrices which can be used in the `ics` and `ics2` functions. Furthermore, there are also two tests for multinormality.

Author(s)

Klaus Nordhausen, Hannu Oja and David E. Tyler

Maintainer: Klaus Nordhausen, `<klaus.nordhausen@tuwien.ac.at>`

`coef.ics` *Extracting the Unmixing Matrix*

Description

Function to extract the unmixing matrix of an class `ics` object.

Usage

```
## S4 method for signature 'ics'
coef(object)
```

Arguments

`object`	object of class `ics`.

Value

The unmixing matrix of an class `ics` object.

Author(s)

Klaus Nordhausen

See Also

`ics-class` and `ics`

`cov4` *Scatter Matrix based on Fourth Moments*

Description

Estimates the scatter matrix based on the 4th moments of the data.

Usage

```
cov4(X, location = "Mean", na.action = na.fail)
```

Arguments

`X`	numeric data matrix or dataframe, missing values are not allowed.
`location`	can be either `Mean`, `Origin` or numeric. If numeric the matrix is computed wrt to the given location.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

If location is `Mean` the scatter matrix of 4th moments is computed wrt to the sample mean. For location = `Origin` it is the scatter matrix of 4th moments wrt to the origin. The scatter matrix is standardized in such a way to be consistent for the regular covariance matrix at the multinormal model. It is given for $n \times p$ matrix X by

$$\frac{1}{p+2} ave_i\{[(x_i - \bar{x})S^{-1}(x_i - \bar{x})'](x_i - \bar{x})'(x_i - \bar{x})\},$$

where $\bar{x}$ is the mean vector and S the regular covariance matrix.

Value

A matrix.

Author(s)

Klaus Nordhausen

References

Cardoso, J.F. (1989), Source separation using higher order moments, in *Proc. IEEE Conf. on Acoustics, Speech and Signal Processing (ICASSP'89)*, 2109–2112. <doi:10.1109/ICASSP.1989.266878>.

Oja, H., Sirkiä, S. and Eriksson, J. (2006), Scatter matrices and independent component analysis, *Austrian Journal of Statistics*, **35**, 175–189.

Examples

```
set.seed(654321)
cov.matrix <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2), ncol=3)
X <- rmvnorm(100, c(0,0,0), cov.matrix)
cov4(X)
cov4(X, location="Origin")
rm(.Random.seed)
```

---

cov4.wt *Weighted Scatter Matrix based on Fourth Moments*

Description

Estimates the weighted scatter matrix based on the 4th moments of the data.

Usage

```
cov4.wt(x, wt = rep(1/nrow(x), nrow(x)), location = TRUE,
        method = "ML", na.action = na.fail)
```

Arguments

x	numeric data matrix or dataframe.
wt	numeric vector of non-negative weights. At least some weights must be larger than zero.
location	TRUE if the weighted location vector should be computed. FALSE when taken wrt to the origin. If numeric the matrix is computed wrt to the given location.
method	Either ML or unbiased. Will be passed on to cov.wt when the Mahalanobis distance is computed.
na.action	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

If location = TRUE, then the scatter matrix is given for a $n \times p$ data matrix X by

$$\frac{1}{p+2} ave_i\{w_i[(x_i - \bar{x}_w)S_w^{-1}(x_i - \bar{x}_w)'](x_i - \bar{x}_w)'(x_i - \bar{x}_w)\},$$

where w_i are the weights standardized such that $\sum w_i = 1$, $\bar{x}_w$ is the weighted mean vector and S_w the weighted covariance matrix. For details about the weighted mean vector and weighted covariance matrix see cov.wt.

Value

A matrix.

Author(s)

Klaus Nordhausen

See Also

cov4, cov.wt

Examples

```
cov.matrix.1 <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2), ncol=3)
X.1 <- rmvnorm(100, c(0,0,0), cov.matrix.1)
cov.matrix.2 <- diag(1,3)
X.2 <- rmvnorm(50, c(1,1,1), cov.matrix.2)
X <- rbind(X.1, X.2)

cov4.wt(X, rep(c(0,1), c(100,50)))
cov4.wt(X, rep(c(1,0), c(100,50)))
```

---

covAxis *One step Tyler Shape Matrix*

Description

This matrix can be used to get from ics the principal axes which is then known as principal axis analysis.

Usage

```
covAxis(X, na.action = na.fail)
```

Arguments

X	numeric data matrix or dataframe.
na.action	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

The `covAxis` matrix V is a given for a sample of size n as

$$p\, ave_i\{[(x_i - \bar{x})S^{-1}(x_i - \bar{x})']^{-1}(x_i - \bar{x})'(x_i - \bar{x})\},$$

where $\bar{x}$ is the mean vector and S the regular covariance matrix.

`covAxis` can be used to perform a Prinzipal Axis Analysis (Critchley et al. 2006) using the function ics. In that case for a centered data matrix X `covAxis` can be used as S2 in ics, where S1 should be in that case the regular covariance matrix.

Value

Matrix of the estimated scatter.

Author(s)

Klaus Nordhausen

References

Critchley , F., Pires, A. and Amado, C. (2006), Principal axis analysis, Technical Report, ***06/14***, *The Open University Milton Keynes.*

Tyler, D.E., Critchley, F., Dümbgen, L. and Oja, H. (2009), Invariant co-ordinate selecetion, Journal of the Royal Statistical Society,*Series B*, ***71***, *549–592. <doi:10.1111/j.1467-9868.2009.00706.x>.*

See Also

ics

Examples

```
data(iris)
iris.centered <- sweep(iris[,1:4], 2, colMeans(iris[,1:4]), "-")
iris.paa <- ics(iris.centered, cov, covAxis, stdKurt = FALSE)
summary(iris.paa)
plot(iris.paa, col=as.numeric(iris[,5]))
mean(iris.paa@gKurt)
emp.align <- iris.paa@gKurt
emp.align

screeplot(iris.paa)
abline(h = 1)
```

---

covOrigin *Covariance Matrix with Respect to the Origin*

Description

Estimates the covariance matrix with respect to the origin.

Usage

```
covOrigin(X, location = NULL, na.action = na.fail)
```

Arguments

X	a numeric data matrix or dataframe.
location	optional location value which serves then as the center instead of the origin.
na.action	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

The covariance matrix S_0 with respect to origin is given for a matrix X with n observations by

$$S_0 = \frac{1}{n} X'X.$$

Value

A matrix.

Author(s)

Klaus Nordhausen

See Also

cov

Examples

```
set.seed(654321)
cov.matrix <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2), ncol=3)
X <- rmvnorm(100,c(0,0,0),cov.matrix)
covOrigin(X)
rm(.Random.seed)
```

---

fitted.ics *Fitted Values of an ICS Object*

Description

Function to compute the fitted values of a `ics` object.

Usage

```
## S4 method for signature 'ics'
fitted(object,index=NULL)
```

Arguments

object	object of class ics.
index	A vector which defines which components should be used to compute the fitted values. The default NULL uses all components.

Value

Returns a dataframe with the fitted values.

Author(s)

Klaus Nordhausen

See Also

ics-class and ics

Examples

```
set.seed(123456)
X1 <- rmvnorm(250, rep(0,8), diag(c(rep(1,6),0.04,0.04)))
X2 <- rmvnorm(50, c(rep(0,6),2,0), diag(c(rep(1,6),0.04,0.04)))
X3 <- rmvnorm(200, c(rep(0,7),2), diag(c(rep(1,6),0.04,0.04)))

X.comps <- rbind(X1,X2,X3)
A <- matrix(rnorm(64),nrow=8)
X <- X.comps %*% t(A)

ics.X.1 <- ics(X)
fitted(ics.X.1)
fitted(ics.X.1,index=c(1,2,3,6,7,8))

rm(.Random.seed)
```

---

ics *Two Scatter Matrices ICS Transformation*

Description

This function implements the two scatter matrices transformation to obtain an invariant coordinate sytem or independent components, depending on the underlying assumptions.

Usage

```
ics(X, S1 = cov, S2 = cov4, S1args = list(), S2args = list(),
    stdB = "Z", stdKurt = TRUE, na.action = na.fail)
```

Arguments

`X`	numeric data matrix or dataframe.
`S1`	name of the first scatter matrix function or a scatter matrix. Default is the regular covariance matrix.
`S2`	name of the second scatter matrix or a scatter matrix. Default is the covariance matrix based on forth order moments. Note that the type of S2 must be the same as S1.
`S1args`	list with optional additional arguments for S1. Only considered if S1 is a function.
`S2args`	list with optional additional arguments for S2. Only considered if S2 is a function.
`stdB`	either "B" or "Z". Defines the way to standardize the matrix B. Default is "Z". Details are given below.
`stdKurt`	Logical, either "TRUE" or "FALSE". Specifies weather the product of the kurtosis values is 1 or not.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

Seeing this function as a tool for data transformation the result is an invariant coordinate selection which can be used for testing and estimation. And if needed the results can be easily retransformed to the original scale. It is possible to use it also for dimension reduction, finding outliers or when searching for clusters in the data. The function can, however, also be used in a modelling framework. In this case it is assumed that the data were created by mixing independent components which have different kurtosis values. If the two scatter matrices used have then the so-called independence property the function can recover the independent components by estimating the unmixing matrix.

By default S1 is the regular covariance matrix `cov` and S2 the matrix of fourth moments `cov4`. However those can be replaced with any other scatter matrix the user prefers. The package **ICS** offers for example also `cov4.wt`, `covAxis`, `covOrigin` or `tM` and in the **ICSNP** are for example further scatters as `duembgen.shape`, `tyler.shape`, `HR.Mest` or `HP1.shape`. But of course also scatters from any other package can be used.

Note that when function names are submitted, the function should return only a scatter matrix. If the function returns more, the scatter should be computed in advance or a wrapper written that yields the required output. For example `tM` returns a list with four elements where the scatter estimate is called V. A simple wrapper would then be `my.tm <- function(x, ...) tM(x, ...)$V`.

For a given choice of S1 and S2 the general idea of the `ics` function is to find the unmixing matrix B and the invariant coordinates (independent coordinates) Z in such a way, that:

(i) The elements of Z are standardized with respect to S1 (S1(Z)=I).

(ii) The elements of Z are uncorrelated with respect to S2. (S2(Z)=D, where D is a diagonal matrix).

(iii) The elements of Z are ordered according to their kurtosis.

Given those criteria, B is unique up to sign changes of its rows. The function provides two options to decide the exact form of B.

(i) Method 'Z' standardizes B such, that all components are right skewed. The criterion used, is the sign of each componentwise difference of mean vector and transformation retransformation median. This standardization is prefered in an invariant coordinate framework.

(ii) Method 'B' standardizes B independent of Z such that the maximum element per row is positive and each row has norm 1. Usual way in an independent component analysis framework.

In principal if S1 and S2 are true scatter matrices the order does not matter. It will just reverse and invert the kurtosis value vector. This is however not true when not both of them are scatter matrices but one or both are shape matrices. In this case the order of the kurtosis values is also reversed, the ratio however then is not 1 but only constant. This is due to the fact that when shape matrices are used, the kurtosis values are only relative ones. Therefore by the default the kurtosis values are standardized such that their product is 1. If no standardization is wanted, the 'stdKurt' argument should be used.

Value

an object of class `ics`.

Author(s)

Klaus Nordhausen

References

Tyler, D.E., Critchley, F., Dümbgen, L. and Oja, H. (2009), Invariant co-ordinate selecetion, Journal of the Royal Statistical Society,*Series B*, ***71***, *549–592. <doi:10.1111/j.1467-9868.2009.00706.x>.*

Oja, H., Sirkiä, S. and Eriksson, J. (2006), Scatter matrices and independent component analysis, Austrian Journal of Statistics, ***35***, *175–189.*

Nordhausen, K., Oja, H. and Tyler, D.E. (2008), Tools for exploring multivariate data: The package ICS, Journal of Statistical Software, ***28***, *1–31. <doi:10.18637/jss.v028.i06>.*

See Also

ICS-package

Examples

```
# example using two functions
set.seed(123456)
X1 <- rmvnorm(250, rep(0,8), diag(c(rep(1,6),0.04,0.04)))
X2 <- rmvnorm(50, c(rep(0,6),2,0), diag(c(rep(1,6),0.04,0.04)))
X3 <- rmvnorm(200, c(rep(0,7),2), diag(c(rep(1,6),0.04,0.04)))

X.comps <- rbind(X1,X2,X3)
A <- matrix(rnorm(64),nrow=8)
X <- X.comps %*% t(A)

ics.X.1 <- ics(X)
summary(ics.X.1)
plot(ics.X.1)
```

```
# compare to
pairs(X)
pairs(princomp(X,cor=TRUE)$scores)

# slow:

# library(ICSNP)
# ics.X.2 <- ics(X, tyler.shape, duembgen.shape, S1args=list(location=0))
# summary(ics.X.2)
# plot(ics.X.2)

rm(.Random.seed)

# example using two computed scatter matrices for outlier detection

library(robustbase)
ics.wood<-ics(wood,tM(wood)$V,tM(wood,2)$V)
plot(ics.wood)

# example using three pictures
library(pixmap)

fig1 <- read.pnm(system.file("pictures/cat.pgm", package = "ICS")[1])
fig2 <- read.pnm(system.file("pictures/road.pgm", package = "ICS")[1])
fig3 <- read.pnm(system.file("pictures/sheep.pgm", package = "ICS")[1])

p <- dim(fig1@grey)[2]

fig1.v <- as.vector(fig1@grey)
fig2.v <- as.vector(fig2@grey)
fig3.v <- as.vector(fig3@grey)
X <- cbind(fig1.v,fig2.v,fig3.v)

set.seed(4321)
A <- matrix(rnorm(9), ncol = 3)
X.mixed <- X %*% t(A)

ICA.fig <- ics(X.mixed)

par.old <- par()
par(mfrow=c(3,3), omi = c(0.1,0.1,0.1,0.1), mai = c(0.1,0.1,0.1,0.1))

plot(fig1)
plot(fig2)
plot(fig3)

plot(pixmapGrey(X.mixed[,1],ncol=p))
plot(pixmapGrey(X.mixed[,2],ncol=p))
plot(pixmapGrey(X.mixed[,3],ncol=p))

plot(pixmapGrey(ics.components(ICA.fig)[,1],ncol=p))
plot(pixmapGrey(ics.components(ICA.fig)[,2],ncol=p))
```

```
plot(pixmapGrey(ics.components(ICA.fig)[,3],ncol=p))

par(par.old)
rm(.Random.seed)
```

---

ics-class *Class ICS*

Description

A S4 class to store results from an invariant coordinate system transformation or independent component computation based on two scatter matrices.

Objects from the Class

Objects can be created by calls of the form `new("ics", ...)`. But usually objects are created by the function `ics`.

Slots

`gKurt`: Object of class `"numeric"`. Gives the generalized kurtosis measures of the components

`UnMix`: Object of class `"matrix"`. The unmixing matrix.

`S1`: Object of class `"matrix"`. The first scatter matrix.

`S2`: Object of class `"matrix"`. The second scatter matrix.

`S1name`: Object of class `"character"`. Name of the first scatter matrix.

`S2name`: Object of class `"character"`. Name of the second scatter matrix.

`Scores`: Object of class `"data.frame"`. The underlying components in the invariant coordinate system.

`DataNames`: Object of class `"character"`. Names of the original variables.

`StandardizeB`: Object of class `"character"`. Names standardization method for UnMix.

`StandardizegKurt`: Object of class `"logical"`. States wether the generalized kurtosis is standardized or not.

Methods

For this class the following generic functions are available: `print.ics`, `summary.ics`, `coef.ics`, `fitted.ics` and `plot.ics`

Note

In case no extractor function for the slots exists, the component can be extracted the usual way using '@'.

Author(s)

Klaus Nordhausen

See Also

`ics`

`ics.components` *Extracting ICS Components*

Description

Function to extract the ICS components of a `ics` object.

Usage

```
ics.components(object)
```

Arguments

`object`	object of class `ics`.

Value

Dataframe that contains the components.

Author(s)

Klaus Nordhausen

See Also

`ics-class` and `ics`

ics2 — *Two Scatter Matrices ICS Transformation Augmented by Two Location Estimates*

Description

This function implements the two scatter matrices transformation to obtain an invariant coordinate sytem or independent components, depending on the underlying assumptions. Differently to `ics` here are also two location functionals used to fix the signs of the components and to get a measure of skewness.

Usage

```
ics2(X, S1 = MeanCov, S2 = Mean3Cov4, S1args = list(), S2args = list(),
     na.action = na.fail)
```

Arguments

`X`	numeric data matrix or dataframe.
`S1`	name of the function which returns the first location vector T1 and scatter matrix S1. Can be also a list which has these values already computed. See details for more information. Default is `MeanCov`.
`S2`	name of the function which returns the second location vector T2 and scatter matrix S2. Can be also a list which has these values already computed. See details for more information. Default is `Mean3Cov4`.
`S1args`	list with optional additional arguments when calling function S1.
`S2args`	list with optional additional arguments when calling function S2.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

For a general discussion about ICS see the help for `ics`. The difference to `ics` is that `S1` and `S2` are either functions which return a list containing a multivariate location and scatter computed on `X` or lists containing these measures computed in advance. Of importance for the resulting lists is that in both cases the location vector is the first element of the list and the scatter matrix the second element. This means most multivariate location - scatter functions can be used directly without the need to write a wrapper.

The invariant coordinates Z are then computed such that (i) T1(Z) = 0, the origin. (ii) S1(Z) = I_p, the identity matrix. (iii) T2(Z) = S, where S is a vector having positive elements and can be seen as a generalized skewness measure (gSkew). (iv) S2(Z) = D, a diagonal matrix with descending elements which can be seen as a generalized kurtosis measure (gKurt).

Hence in this function there are no options to standardize Z or the transformation matrix B as everything is specified by `S1` and `S2`.

Note also that `ics2` makes hardly any input checks.

Value

an object of class `ics2` inheriting from class `ics`.

Author(s)

Klaus Nordhausen

References

Tyler, D.E., Critchley, F., DÃ¼mbgen, L. and Oja, H. (2009), *Invariant co-ordinate selecetion*, Journal of the Royal Statistical Society,*Series B*, **71**, *549–592. <doi:10.1111/j.1467-9868.2009.00706.x>.*

Nordhausen, K., Oja, H. and Ollila, E. (2011), Multivariate Models and the First Four Moments, In Hunter, D.R., Richards, D.S.R. and Rosenberger, J.L. (editors) "Nonparametric Statistics and Mixture Models: A Festschrift in Honor of Thomas P. Hettmansperger", 267–287, World Scientific, Singapore. <doi:10.1142/9789814340564_0016>.

See Also

`ics`

Examples

```
set.seed(123456)
X1 <- rmvnorm(250, rep(0,8), diag(c(rep(1,6),0.04,0.04)))
X2 <- rmvnorm(50, c(rep(0,6),2,0), diag(c(rep(1,6),0.04,0.04)))
X3 <- rmvnorm(200, c(rep(0,7),2), diag(c(rep(1,6),0.04,0.04)))

X.comps <- rbind(X1,X2,X3)
A <- matrix(rnorm(64),nrow=8)
X <- X.comps %*% t(A)

# the default
ics2.X.1 <- ics2(X2)
summary(ics2.X.1)

# using another function as S2 not with its default
ics2.X.2 <- ics2(X2, S2 = tM, S2args = list(df = 2))
summary(ics2.X.2)

# computing in advance S2 and using another S1
Scauchy <- tM(X)
ics2.X.2 <- ics2(X2, S1 = tM, S2 = Scauchy, S1args = list(df = 5))
summary(ics2.X.2)
plot(ics2.X.2)
```

ics2-class *Class ICS2*

Description

A S4 class to store results from an invariant coordinate system transformation or independent component computation based on two scatter matrices and two location vectors.

Objects from the Class

Objects can be created by calls of the form `new("ics2", ...)`. But usually objects are created by the function `ics2`. The Class inherits from the `ics` class.

Slots

`gSkew`: Object of class `"numeric"`. Gives the generalized skewness measures of the components

`gKurt`: Object of class `"numeric"`. Gives the generalized kurtosis measures of the components

`UnMix`: Object of class `"matrix"`. The unmixing matrix.

`S1`: Object of class `"matrix"`. The first scatter matrix.

`S2`: Object of class `"matrix"`. The second scatter matrix.

`T1`: Object of class `"numeric"`. The first location vector.

`T2`: Object of class `"numeric"`. The second location vector.

`S1name`: Object of class `"character"`. Name of the first scatter matrix.

`S2name`: Object of class `"character"`. Name of the second scatter matrix.

`S1args`: Object of class `"list"`. Additional arguments needed when calling function S1.

`S2args`: Object of class `"list"`. Additional arguments needed when calling function S2.

`Scores`: Object of class `"data.frame"`. The underlying components in the invariant coordinate system.

`DataNames`: Object of class `"character"`. Names of the original variables.

`StandardizeB`: Object of class `"character"`. Names standardization method for UnMix.

`StandardizegKurt`: Object of class `"logical"`. States wether the generalized kurtosis is standardized or not.

Methods

For this class the following generic functions are available: `print.ics2`, `summary.ics2` But naturally the other methods like plot, coef, fitted and so from class ics work via inheritance.

Note

In case no extractor function for the slots exists, the component can be extracted the usual way using '@'.

Author(s)

Klaus Nordhausen

See Also

ics2

---

mean3 *Location Estimate based on Third Moments*

Description

Estimates the location based on third moments.

Usage

```
mean3(X, na.action = na.fail)
```

Arguments

X	numeric data matrix or dataframe with at least two columns.
na.action	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

This Location Estimate is defined for $n \times p$ matrix X as

$$\frac{1}{p} ave_i\{[(x_i - \bar{x})S^{-1}(x_i - \bar{x})']x_i\},$$

where $\bar{x}$ is the mean vector and S the regular covariance matrix.

Value

A vector.

Author(s)

Klaus Nordhausen

References

Oja, H., Sirkiä, S. and Eriksson, J. (2006), Scatter matrices and independent component analysis, Austrian Journal of Statistics, ***35****, 175–189.*

Examples

```
set.seed(654321)
cov.matrix <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2), ncol=3)
X <- rmvnorm(100, c(0,0,0), cov.matrix)
mean3(X)
rm(.Random.seed)
```

`Mean3Cov4` *Location Vector Based on 3rd Moments and Scatter Matrix Based on 4th Moments*

Description

The function returns for some multivariate data the location vector based on 3rd moments and the scatter matrix based on 4th moments.

Usage

```
Mean3Cov4(x)
```

Arguments

`x`	a numeric data matrix.

Details

Note that the scatter matrix of 4th moments is computed with respect to the mean vector and not with respect to the location vector based on 3rd moments.

Value

A list containing:

`locations`	The locatin vector based on 3rd moments as computed by `mean3`.
`scatter`	The scatter matrix based on 4th moments as computed by `cov4`.

Author(s)

Klaus Nordhausen

See Also

`mean3`, `cov4`

Examples

```
X <- rmvnorm(200, 1:3, diag(2:4))
Mean3Cov4(X)
```

MeanCov — *Mean Vector and Covariance Matrix*

Description

The function returns for some multivariate data the mean vector and covariance matrix.

Usage

```
MeanCov(x)
```

Arguments

`x`	a numeric data matrix.

Value

A list containing:

`locations`	The mean vector as computed by `colMeans`.
`scatter`	The covariance matrix as computed by `cov`.

Author(s)

Klaus Nordhausen

See Also

`colMeans`, `cov`

Examples

```
X <- rmvnorm(200, 1:3, diag(2:4))
MeanCov(X)
```

mvnorm.kur.test — *Test of Multivariate Normality Based on Kurtosis*

Description

Test for multivariate normality which uses as criterion the kurtosis measured by the ratio of regular covariance matrix and matrix of fourth moments.

Usage

```
mvnorm.kur.test(X, method = "integration", n.simu = 1000,
                na.action = na.fail)
```

Arguments

X	a numeric data frame or matrix.
method	defines the method used for the computation of the p-value. The possibilites are "integration" (default), "satterthwaite" or "simulation". Details below.
n.simu	if 'method=simulation' this specifies the number of replications in the simulation.
na.action	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

This test implements the multivariate normality test based on kurtosis measured by two different scatter estimates as described in Kankainen, Taskinen and Oja. The choice here is based on the regular covariance matrix and matrix of fourth moments (cov4). The limiting distribution of the test statistic W is a linear combination of independent chi-square variables with different degrees of freedom. Exact limiting p-values or approximated p-values are obtained by using the function pchisqsum. However Kankainen et al. mention that even for n = 200 the convergence can be poor, therefore also p-values simulated under the NULL can be obtained.

Note that the test statistic used is a symmetric version of the one in the paper to guarantee affine invariance.

Value

A list with class 'htest' containing the following components:

statistic	the value of the test statistic W.
parameter	the degrees of freedom for the test statistic W with their weights or the number of replications depending on the chosen method.
p.value	the p-value for the test.
method	a character string indicating what type of test was performed.
data.name	a character string giving the name of the data.

Author(s)

Klaus Nordhausen

References

Kankainen, A., Taskinen, S. and Oja, H. (2007), Tests of multinormality based on location vectors and scatter matrices, Statistical Methods and Applications, ***16***, *357–379. <doi:10.1007/s10260-007-0045-9>.*

See Also

mvnorm.skew.test

Examples

```
X<-rmvnorm(100, c(2, 4, 5))
mvnorm.kur.test(X)
mvnorm.kur.test(X, method = "satt")
mvnorm.kur.test(X, method = "simu")
```

---

mvnorm.skew.test *Test of Multivariate Normality Based on Skewness*

Description

Test for multivariate normality which uses as criterion the skewness measured as the difference between location estimates based on first respectively third moments

Usage

```
mvnorm.skew.test(X, na.action = na.fail)
```

Arguments

`X`	a numeric data frame or matrix.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

This test implements the multivariate normality test based on skewness measured by two different location estimates as described in Kankainen, Taskinen and Oja. The choice here is based on the regular mean vector and the location estimate based on third moments (`mean3`). The scatter matrix used is the regular covariance matrix.

Value

A list with class 'htest' containing the following components:

`statistic`	the value of the test statistic U.
`parameter`	the degrees of freedom for the statistic U.
`p.value`	the p-value for the test.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name of the data.

Author(s)

Klaus Nordhausen

References

Kankainen, A., Taskinen, S. and Oja, H. (2007),Tests of multinormality based on location vectors and scatter matrices, Statistical Methods and Applications, ***16***, *357–379. <doi:10.1007/s10260-007-0045-9>.*

See Also

mvnorm.kur.test

Examples

```
X<-rmvnorm(100,c(2,4,5))
mvnorm.skew.test(X)
```

---

plot.ics *Scatterplot for a ICS Object*

Description

Scatterplot matrix for a `ics` object.

Usage

```
## S4 method for signature 'ics,missing'
plot(x, index = NULL, ...)
```

Arguments

`x`	object of class `ics`
`index`	index vector of which components should be plotted. See details for further information
`...`	other arguments for `plot`

Details

If no index vector is given the function plots the full scatterplots matrix only if there are less than seven components. Otherwise the three first and three last components will be plotted. This is because the components with extreme kurtosis are the most interesting ones.

Author(s)

Klaus Nordhausen

See Also

screeplot.ics, ics-class and ics

Examples

```
set.seed(123456)
X1 <- rmvnorm(250, rep(0,8), diag(c(rep(1,6),0.04,0.04)))
X2 <- rmvnorm(50, c(rep(0,6),2,0), diag(c(rep(1,6),0.04,0.04)))
X3 <- rmvnorm(200, c(rep(0,7),2), diag(c(rep(1,6),0.04,0.04)))

X.comps <- rbind(X1,X2,X3)
A <- matrix(rnorm(64),nrow=8)
X <- X.comps %*% t(A)

ics.X.1 <- ics(X)
plot(ics.X.1)
plot(ics.X.1,index=1:8)
rm(.Random.seed)
```

---

print.ics *Basic information of ICS Object*

Description

Prints the minimal information of an `ics` object.

Usage

```
## S4 method for signature 'ics'
show(object)
```

Arguments

`object`	object of class `ics`.

Author(s)

Klaus Nordhausen

See Also

`ics-class` and `ics`

print.ics2 *Basic information of ICS2 Object*

Description

Prints the minimal information of an `ics2` object.

Usage

```
## S4 method for signature 'ics2'
show(object)
```

Arguments

`object`	object of class `ics2`.

Author(s)

Klaus Nordhausen

See Also

`ics2-class` and `ics2`

scovq *Supervised scatter matrix based on quantiles*

Description

Function for a supervised scatter matrix that is the weighted covariance matrix of `x` with weights 1/(`q2`-`q1`) if `y` is between the lower (`q1`) and upper (`q2`) quantile and 0 otherwise (or vice versa).

Usage

```
scovq(x, y, q1 = 0, q2 = 0.5, pos = TRUE, type = 7,
      method = "unbiased", na.action = na.fail,
      check = TRUE)
```

Arguments

`x`	numeric data matrix with at least two columns.
`y`	numerical vector specifying the dependent variable.
`q1`	percentage for lower quantile of `y`. With 0 <= `q1` < `q2`. See details.
`q2`	percentage for upper quantile of `y`. With `q1` < `q2` <= 1. See details.
`pos`	logical. If TRUE then the weights are 1/(`q2`-`q1`) if `y` is between the `q1`- and `q2`-quantiles and 0 othervise. If FALSE then the weights are 0 if `y` between `q1`- and `q2`-quantiles and 1/(1-`q2`+`q1`) otherwise.
`type`	passed on to function `quantile`.
`method`	passed on to function `cov.wt`.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.
`check`	logical. Checks if the input should be checked for consistency. If not needed setting it to FALSE might save some time.

Details

The weights for this supervised scatter matrix for `pos=TRUE` are $w(y) = I(q1 - quantile < y < q2 - quantile)/(q2 - q1)$. Then `scovq` is calculated as

$$scovq = \sum w(y)(x - \bar{x}_w)'(x - \bar{x}_w).$$

where $\bar{x}_w = \sum w(y)x$.

To see how this function can be used in the context of supervised invariant coordinate selection see the example below.

Value

a matrix.

Author(s)

Klaus Nordhausen

References

Liski, E., Nordhausen, K. and Oja, H. (2014), Supervised invariant coordinate selection, Statistics: A Journal of Theoretical and Applied Statistics, ***48****, 711–731. <doi:10.1080/02331888.2013.800067>.*

See Also

`cov.wt` and `ics`

Examples

```
# Creating some data

# The number of explaining variables
p <- 10
# The number of observations
n <- 400
# The error variance
sigma <- 0.5
# The explaining variables
X <- matrix(rnorm(p*n),n,p)
# The error term
epsilon <- rnorm(n, sd = sigma)
# The response
y <- X[,1]^2 + X[,2]^2*epsilon


# SICS with ics

X.centered <- sweep(X,2,colMeans(X),"-")
SICS <- ics(X.centered, S1=cov, S2=scovq, S2args=list(y=y, q1=0.25,
        q2=0.75, pos=FALSE), stdKurt=FALSE, stdB="Z")

# Assuming it is known that k=2, then the two directions
# of interest are choosen as:

k <- 2
KURTS <- SICS@gKurt
KURTS.max <- ifelse(KURTS >= 1, KURTS, 1/KURTS)
ordKM <- order(KURTS.max, decreasing = TRUE)

indKM <- ordKM[1:k]

# The two variables of interest
Zk <- ics.components(SICS)[,indKM]

# The correspondings transformation matrix
Bk <- coef(SICS)[indKM,]

# The corresponding projection matrix
Pk <- t(Bk) %*% solve(Bk %*% t(Bk)) %*% Bk

# Visualization
pairs(cbind(y,Zk))

# checking the subspace difference

# true projection

B0 <- rbind(rep(c(1,0),c(1,p-1)),rep(c(0,1,0),c(1,1,p-2)))
P0 <- t(B0) %*% solve(B0 %*% t(B0)) %*% B0
```

```
# crone and crosby subspace distance measure, should be small
k - sum(diag(P0 %*% Pk))
```

---

screeplot.ics *Screeplot for an ICS Object*

Description

Plots the kurtosis measures of an `ics` object against its index number. Two versions of this screeplot are available.

Usage

```
## S3 method for class 'ics'
screeplot(x, index = NULL, type = "barplot",
          main = deparse(substitute(x)), ylab = "generalized kurtosis",
          xlab = "component", names.arg = index, labels = TRUE, ...)
```

Arguments

`x`	object of class `ics`.
`index`	index of the components to be plottes. If NULL all components are used.
`type`	`barplot` if a barplot or `lines` if a line plot is preferred.
`main`	main title of the plot.
`ylab`	y-axis label.
`xlab`	x-axis label.
`names.arg`	`names.arg` argument passed on to `barplot`.
`labels`	`labels` argument for the labels of the x-axis passed on to `axis`.
`...`	other arguments for the plotting functions.

Author(s)

Klaus Nordhausen

See Also

`plot.ics`, `ics-class` and `ics`

Examples

```
set.seed(654321)
A <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2),ncol=3)
eigen.A <- eigen(A)
sqrt.A <- eigen.A$vectors %*% (diag(eigen.A$values))^0.5 %*% t(eigen.A$vectors)
normal.ic <- cbind(rnorm(800), rnorm(800), rnorm(800))
mix.ic <- cbind(rt(800,4), rnorm(800), runif(800,-2,2))

data.normal <- normal.ic %*% t(sqrt.A)
data.mix <- mix.ic %*% t(sqrt.A)

par(mfrow=c(1,2))
screeplot(ics(data.normal))
screeplot(ics(data.mix), type="lines")
par(mfrow=c(1,1))
rm(.Random.seed)

screeplot(ics(data.normal), names.arg=paste("IC", 1:ncol(A), sep=""), xlab="")
```

---

summary.ics — *Summarize a ICS object*

Description

Summarizes and prints a `ics` object in an informative way.

Usage

```
## S4 method for signature 'ics'
summary(object, digits = 4)
```

Arguments

`object`	object of class `ics`.
`digits`	number of digits for the numeric output.

Author(s)

Klaus Nordhausen

See Also

`ics-class` and `ics`

summary.ics2 *Summarize a ICS2 object*

Description

Summarizes and prints a ics2 object in an informative way.

Usage

```
## S4 method for signature 'ics2'
summary(object, digits = 4)
```

Arguments

object	object of class ics2.
digits	number of digits for the numeric output.

Author(s)

Klaus Nordhausen

See Also

ics2-class and ics2

tM *Joint M-estimation of Location and Scatter for a Multivariate t-distribution*

Description

The functions implements three EM algorithms to M-estimate the location vector and scatter matrix of a multivariate t-distribution.

Usage

```
tM(X, df = 1, alg = "alg3", mu.init = NULL, V.init = NULL,
        gamma.init = NULL, eps = 1e-06, maxiter = 100,
        na.action = na.fail)
```

Arguments

`X`	numeric data matrix or dataframe.
`df`	assumed degrees of freedom of the t-distribution. Default is `1` which corresponds to the Cauchy distribution.
`alg`	specifies which algorithm to use. Options are `alg1`, `alg2` or `alg3`. `alg3` is the default.
`mu.init`	initial value for the location vector if available.
`V.init`	initial value for the scatter matrix if available.
`gamma.init`	initial value for gamma if available. Only needed for `alg2`.
`eps`	convergence tolerance.
`maxiter`	maximum number of iterations.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.

Details

This implements the EM algorithms described in Kent et al. (1994). The norm used to define convergence is as in Arslan et al. (1995).

Algorithm 1 is valid for all degrees of freedom `df` > 0. Algorithm 2 is well defined only for degrees of freedom `df` > 1. Algorithm 3 is the limiting case of Algorithm 2 with degrees of freedom `df` = 1.

The performance of the algorithms are compared in Arslan et al. (1995).

Note that `cov.trob` in the MASS package implements also a covariance estimate for a multivariate t-distribution. That function provides for example also the possibility to fix the location. It requires however that the degrees of freedom exceeds 2.

Value

A list containing:

`mu`	vector with the estimated loaction.
`V`	matrix of the estimated scatter.
`gam`	estimated value of gamma. Only present when `alg2` is used.
`iter`	number of iterations.

Author(s)

Klaus Nordhausen

References

Kent, J.T., Tyler, D.E. and Vardi, Y. (1994), A curious likelihood identity for the multivariate t-distribution, Communications in Statistics, Simulation and Computation, ***23****, 441–453. <doi:10.1080/03610919408813180>.*

Arslan, O., Constable, P.D.L. and Kent, J.T. (1995), Convergence behaviour of the EM algorithm for the multivariate t-distribution, Communications in Statistics, Theory and Methods, ***24****, 2981–3000. <doi:10.1080/03610929508831664>.*

See Also

cov.trob

Examples

```
set.seed(654321)
cov.matrix <- matrix(c(3,2,1,2,4,-0.5,1,-0.5,2), ncol=3)
X <- rmvt(100, cov.matrix, 1)
tM(X)
rm(.Random.seed)
```

Index