Package: ICS (via r-universe)

October 15, 2024

Type Package

Title Tools for Exploring Multivariate Data via ICS/ICA

Version 1.4-1

Date 2023-09-17

Author Klaus Nordhausen [aut, cre] (<https://orcid.org/0000-0002-3758-8501>), Andreas Alfons [aut] (<https://orcid.org/0000-0002-2513-3788>), Aurore Archimbaud [aut] (<https://orcid.org/0000-0002-6511-9091>), Hannu Oja [aut] (<https://orcid.org/0000-0002-4945-5976>), Anne Ruiz-Gazen [aut] (<https://orcid.org/0000-0001-8970-8061>), David E. Tyler [aut]

Maintainer Klaus Nordhausen <klausnordhausenR@gmail.com>

Depends R (>= 2.5.0), methods, mvtnorm

Imports survey, graphics

Suggests pixmap, robustbase, MASS, ICSNP, testthat (>= 3.0.0), ICSOutlier

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, <https://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.

License GPL (>= 2)

LazyLoad yes

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.2.3

Config/testthat/edition 3

Repository CRAN

Date/Publication 2023-09-21 07:10:02 UTC

Contents

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

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, <https://www.ajs.or.at/index.php/ajs/article/view/vol35,%2C
%207>) method of two different scatter matrices to obtain an invariant coordinate system or independent components, depending on the underlying assumptions.

Details

Package:	ICS
Type:	Package
Title:	Tools for Exploring Multivariate Data via ICS/ICA
Version:	1.4-1
Date:	2023-09-17
Authors@R:	c(person("Klaus", "Nordhausen", email = "klausnordhausenR@gmail.com", role = c("aut", "cre"), c
Author:	Klaus Nordhausen [aut, cre] (<https://orcid.org/0000-0002-3758-8501>), Andreas Alfons [aut] (<htt
Maintainer:	Klaus Nordhausen <klausnordhausenR@gmail.com>
Depends:	R (>= 2.5.0), methods, mvtnorm
Imports:	survey, graphics
Suggests:	pixmap, robustbase, MASS, ICSNP, testthat (>= 3.0.0), ICSOutlier
Description:	Implementation of Tyler, Critchley, Duembgen and Oja's (JRSS B, 2009, <doi:10.1111/j.1467-9868.
License:	GPL (>= 2)
LazyLoad:	yes
Encoding:	UTF-8
NeedsCompilation:	no
Roxygen:	list(markdown = TRUE)
RoxygenNote:	7.2.3
Config/testthat/edition:	3

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. Besides, the function `ICS` provides these package tools to work with objects of this class, and some scatter matrices which can be used in the `ICS` function. Furthermore, there are also two tests for multinormality. Note that starting with version 1.4-0 the functions `ics` and `ics2` are not recommended anymore and everything can be done in a more efficient way using the function `ICS` which combines the functionality of the original two functions and also provides an improved algorithm for certain scatter combinations. Furthermore, does ICS return an S3 object and not anymore S4 objects as ics and ics2 did. In the long run functions ics and ics2 will be removed from the package.

Index of help topics:

```
ICS-S3                  Two Scatter Matrices ICS Transformation
ICS-package             Tools for Exploring Multivariate Data via
                        ICS/ICA
ICS_scatter             Location and Scatter Estimates for ICS
Mean3Cov4               Location Vector Based on 3rd Moments and
                        Scatter Matrix Based on 4th Moments
MeanCov                 Mean Vector and Covariance Matrix
coef.ICS-S3             To extract the Coefficient Matrix of the ICS
                        Transformation
coef.ics                To extract the Unmixing Matrix
components              To extract the Component Scores of the ICS
                        Transformation
cov4                    Scatter Matrix based on Fourth Moments
cov4.wt                 Weighted Scatter Matrix based on Fourth Moments
covAxis                 One step Tyler Shape Matrix
covOrigin               Covariance Matrix with Respect to the Origin
covW                    One-step M-estimator
fitted.ICS-S3           Fitted Values of the ICS Transformation
fitted.ics              Fitted Values of an ICS Object
gen_kurtosis            To extract the Generalized Kurtosis Values of
                        the ICS Transformation
ics                     Two Scatter Matrices ICS Transformation
ics-class               Class ICS
ics.components          Extracting ICS Components
ics2                    Two Scatter Matrices ICS Transformation
                        Augmented by Two Location Estimates
ics2-class              Class ICS2
mean3                   Location Estimate based on Third Moments
mvnorm.kur.test         Test of Multivariate Normality Based on
                        Kurtosis
mvnorm.skew.test        Test of Multivariate Normality Based on
                        Skewness
plot.ICS-S3             Scatterplot Matrix of Component Scores from the
                        ICS Transformation
plot.ics                Scatterplot for a ICS Object
print.ICS-S3            Basic information of ICS Object
print.ics               Basic information of ICS Object
print.ics2              Basic information of ICS2 Object
scovq                   Supervised scatter matrix based on quantiles
screeplot.ICS-S3        Screeplot for an 'ICS' Object
screeplot.ics           Screeplot for an ICS Object
summary.ICS-S3          To summarize an 'ICS' object
summary.ics             To summarize an ICS object
summary.ics2            To summarize an ICS2 object
tM                      Joint M-estimation of Location and Scatter for
```

```
a Multivariate t-distribution
```

Author(s)

Klaus Nordhausen [aut, cre] (<https://orcid.org/0000-0002-3758-8501>), Andreas Alfons [aut] (<https://orcid.org/0000-0002-2513-3788>), Aurore Archimbaud [aut] (<https://orcid.org/0000-0002-6511-9091>), Hannu Oja [aut] (<https://orcid.org/0000-0002-4945-5976>), Anne Ruiz-Gazen [aut] (<https://orcid.org/0000-0001-8970-8061>), David E. Tyler [aut]

Maintainer: Klaus Nordhausen <klausnordhausenR@gmail.com>

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>.*

Archimbaud, A., Drmac, Z., Nordhausen, K., Radojicic, U. and Ruiz-Gazen, A. (2023), Numerical considerations and a new implementation for ICS, SIAM Journal on Mathematics of Data Science, ***5***, *97–121. <doi:10.1137/22M1498759>.*

---

coef.ics — *To extract the Unmixing Matrix*

Description

Extracts the unmixing matrix of a class `ics` object.

Usage

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

Arguments

`object`	object of class `ics`.

Value

The unmixing matrix of a class `ics` object.

Author(s)

Klaus Nordhausen

See Also

`ics-class` and `ics`

coef.ICS-S3 *To extract the Coefficient Matrix of the ICS Transformation*

Description

Extracts the coefficient matrix of a linear transformation to an invariant coordinate system. Each row of the matrix contains the coefficients of the transformation to the corresponding component.

Usage

```
## S3 method for class 'ICS'
coef(object, select = NULL, drop = FALSE, index = NULL, ...)
```

Arguments

`object`	an object inheriting from class `"ICS"` containing results from an ICS transformation.
`select`	an integer, character, or logical vector specifying for which components to extract the coefficients, or `NULL` to extract the coefficients for all components.
`drop`	a logical indicating whether to return a vector rather than a matrix in case coefficients are extracted for a single component (default to `FALSE`).
`index`	an integer vector specifying for which components to extract the coefficients, or `NULL` to extract coefficients for all components. Note that `index` is deprecated and may be removed in the future, use `select` instead.
`...`	additional arguments are ignored.

Value

A numeric matrix or vector containing the coefficients for the requested components.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`gen_kurtosis()`, `components()`, `fitted()`, and `plot()` methods

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
coef(out)
coef(out, select = c(1,4))
coef(out, select = 1, drop = FALSE)
```

`components` *To extract the Component Scores of the ICS Transformation*

Description

Extracts the components scores of an invariant coordinate system obtained via an ICS transformation.

Usage

```
components(x, ...)

## S3 method for class 'ICS'
components(x, select = NULL, drop = FALSE, index = NULL, ...)
```

Arguments

`x`	an object inheriting from class `"ICS"` containing results from an ICS transformation.
`...`	additional arguments to be passed down.
`select`	an integer, character, or logical vector specifying which components to extract, or `NULL` to extract all components.
`drop`	a logical indicating whether to return a vector rather than a matrix in case a single component is extracted (default to `FALSE`).
`index`	an integer vector specifying which components to extract, or `NULL` to extract all components. Note that `index` is deprecated and may be removed in the future, use `select` instead.

Value

A numeric matrix or vector containing the requested components.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`gen_kurtosis()`, `coef()`, `fitted()`, and `plot()` methods

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
components(out)
components(out, select = c(1,4))
components(out, select = 1, drop = FALSE)
```

`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 containing the estimated fourth moments scatter.

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., Sirki?, 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 containing the estimated weighted fourth moments scatter.

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 the principal axes from ics, 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 $n \times p$ data matrix X 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

A matrix containing the estimated one step Tyler shape matrix.

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., D?mbgen, 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 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 containing the estimated covariance matrix with respect to the origin.

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)
```

`covW` *One-step M-estimator*

Description

Estimates the scatter matrix based on one-step M-estimator using mean and covariance matrix as starting point.

Usage

```
covW(X, na.action = na.fail, alpha = 1, cf = 1)
```

Arguments

`X`	numeric $n \times p$ data matrix or dataframe.
`na.action`	a function which indicates what should happen when the data contain 'NA's. Default is to fail.
`alpha`	parameter of the one-step M-estimator. By default equals to 1.
`cf`	consistency factor of the one-step M-estimator. By default equals to 1.

Details

It is given for $n \times p$ matrix X by

$$COV_w(X) = \frac{1}{n} cf \sum_{i=1}^{n} w(D^2(x_i))(x_i - \bar{x})^\top (x_i - \bar{x}),$$

where $\bar{x}$ is the mean vector, $D^2(x_i)$ is the squared Mahalanobis distance, $w(d) = d^\alpha$ is a non-negative and continuous weight function and cf is a consistency factor. Note that the consistency factor, which makes the estimator consistent at the multivariate normal distribution, is in most case unknown and therefore the default is to use simply `cf = 1`.

- If $w(d) = 1$, we get the covariance matrix `cov()` (up to the factor $1/(n-1)$ instead of $1/n$).
- If $\alpha = -1$, we get the `covAxis()`.
- If $\alpha = 1$, we get the `cov4()` with $cf = \frac{1}{p+2}$.

Value

A matrix containing the one-step M-scatter.

Author(s)

Aurore Archimbaud and Klaus Nordhausen

References

Archimbaud, A., Drmac, Z., Nordhausen, K., Radojicic, U. and Ruiz-Gazen, A. (2023). SIAM Journal on Mathematics of Data Science (SIMODS), Vol.5(1):97–121. doi:10.1137/22M1498759.

See Also

`cov()`, `cov4()`, `covAxis()`

Examples

```
data(iris)
X <- iris[,1:4]

# Equivalence with covAxis
covW(X, alpha = -1, cf = ncol(X))
covAxis(X)
```

```
# Equivalence with cov4
covW(X, alpha = 1, cf = 1/(ncol(X)+2))
cov4(X)

# covW with alpha = 0.5
covW(X, alpha = 0.5)
```

`fitted.ics` *Fitted Values of an ICS Object*

Description

Computes the fitted values of an `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)
```

---

fitted.ICS-S3 *Fitted Values of the ICS Transformation*

Description

Computes the fitted values based on an invariant coordinate system obtained via an ICS transformation. When using all components, computing the fitted values constitutes a backtransformation to the observed data. When using fewer components, the fitted values can often be viewed as reconstructions of the observed data with noise removed.

Usage

```
## S3 method for class 'ICS'
fitted(object, select = NULL, index = NULL, ...)
```

Arguments

`object`	an object inheriting from class "ICS" containing results from an ICS transformation.
`select`	an integer, character, or logical vector specifying which components to use for computing the fitted values, or `NULL` to compute the fitted values from all components.
`index`	an integer vector specifying which components to use for computing the fitted values, or `NULL` to compute the fitted values from all components. Note that `index` is deprecated and may be removed in the future, use `select` instead.
`...`	additional arguments are ignored.

Value

A numeric matrix containing the fitted values.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`gen_kurtosis()`, `coef()`, `components()`, and `plot()` methods

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
fitted(out)
fitted(out, select = 4)
```

---

gen_kurtosis *To extract the Generalized Kurtosis Values of the ICS Transformation*

Description

Extracts the generalized kurtosis values of the components obtained via an ICS transformation.

Usage

```
gen_kurtosis(object, ...)

## S3 method for class 'ICS'
gen_kurtosis(object, select = NULL, scale = FALSE, index = NULL, ...)
```

Arguments

`object`	an object inheriting from class `"ICS"` containing results from an ICS transformation.
`...`	additional arguments to be passed down.
`select`	an integer, character, or logical vector specifying for which components to extract the generalized kurtosis values, or `NULL` to extract the generalized kurtosis values of all components.
`scale`	a logical indicating whether to scale the generalized kurtosis values to have product 1 (default to `FALSE`). See 'Details' for more information.
`index`	an integer vector specifying for which components to extract the generalized kurtosis values, or `NULL` to extract the generalized kurtosis values of all components. Note that `index` is deprecated and may be removed in the future, use `select` instead.

Details

The argument `scale` is useful when ICS is performed with shape matrices rather than true scatter matrices. Let S_1 and S_2 denote the scatter or shape matrices used in ICS.

If both S_1 and S_2 are true scatter matrices, their order in principal does not matter. Changing their order will just reverse the order of the components and invert the corresponding generalized kurtosis values.

The same does not hold when at least one of them is a shape matrix rather than a true scatter matrix. In that case, changing their order will also reverse the order of the components, but the ratio of the generalized kurtosis values is no longer 1 but only a constant. This is due to the fact that when shape matrices are used, the generalized kurtosis values are only relative ones. It is then useful to scale the generalized kurtosis values such that their product is 1.

Value

A numeric vector containing the generalized kurtosis values of the requested components.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`coef()`, `components()`, `fitted()`, and `plot()` methods

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
gen_kurtosis(out)
gen_kurtosis(out, scale = TRUE)
gen_kurtosis(out, select = c(1,4))
```

---

ics *Two Scatter Matrices ICS Transformation*

Description

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 test 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, in order to find outliers or clusters in the data. The function can, 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 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`, `covW` or `tM` and the **ICSNP** offers 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 generalized 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 one or both are shape matrices (and not both of them are scatter 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`.

Note

Function `ics()` reached the end of its lifecycle, please use `ICS()` instead. In future versions, `ics()` will be deprecated and eventually removed.

Author(s)

Klaus Nordhausen

References

Tyler, D.E., Critchley, F., D?mbgen, 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., Sirki?, 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`, `ICS`

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-S3 *Two Scatter Matrices ICS Transformation*

Description

Transforms the data via two scatter matrices to an invariant coordinate system or independent components, depending on the underlying assumptions. Function `ICS()` is intended as a replacement for `ics()` and `ics2()`, and it combines their functionality into a single function. Importantly, the results are returned as an S3 object rather than an S4 object. Furthermore, `ICS()` implements recent improvements, such as a numerically stable algorithm based on the QR algorithm for a common family of scatter pairs.

Usage

```
ICS(
  X,
  S1 = ICS_cov,
  S2 = ICS_cov4,
  S1_args = list(),
  S2_args = list(),
  algorithm = c("whiten", "standard", "QR"),
  center = FALSE,
  fix_signs = c("scores", "W"),
  na.action = na.fail
)
```

Arguments

`X`	a numeric matrix or data frame containing the data to be transformed.
`S1`	a numeric matrix containing the first scatter matrix, an object of class `"ICS_scatter"` (that typically contains the location vector and scatter matrix as `location` and `scatter` components), or a function that returns either of those options. The default is function `ICS_cov()` for the sample covariance matrix.
`S2`	a numeric matrix containing the second scatter matrix, an object of class `"ICS_scatter"` (that typically contains the location vector and scatter matrix as `location` and `scatter` components), or a function that returns either of those options. The default is function `ICS_cov4()` for the covariance matrix based on fourth order moments.
`S1_args`	a list containing additional arguments for `S1` (only relevant if `S1` is a function).
`S2_args`	a list containing additional arguments for `S2` (only relevant if `S2` is a function).
`algorithm`	a character string specifying with which algorithm the invariant coordinate system is computed. Possible values are `"whiten"`, `"standard"` or `"QR"`. See 'Details' for more information.

`center`	a logical indicating whether the invariant coordinates should be centered with respect to first locattion or not (default to `FALSE`). Centering is only applicable if the first scatter object contains a location component, otherwise this is set to `FALSE`. Note that this only affects the scores of the invariant components (output component `scores`), but not the generalized kurtosis values (output component `gen_kurtosis`).
`fix_signs`	a character string specifying how to fix the signs of the invariant coordinates. Possible values are `"scores"` to fix the signs based on (generalized) skewness values of the coordinates, or `"W"` to fix the signs based on the coefficient matrix of the linear transformation. See 'Details' for more information.
`na.action`	a function to handle missing values in the data (default to `na.fail`, see its help file for alternatives).

Details

For a given scatter pair S_1 and S_2, a matrix Z (in which the columns contain the scores of the respective invariant coordinates) and a matrix W (in which the rows contain the coefficients of the linear transformation to the respective invariant coordinates) are found such that:

- The columns of Z are whitened with respect to S_1. That is, $S_1(Z) = I$, where I denotes the identity matrix.
- The columns of Z are uncorrelated with respect to S_2. That is, $S_2(Z) = D$, where D is a diagonal matrix.
- The columns of Z are ordered according to their generalized kurtosis.

Given those criteria, W is unique up to sign changes in its rows. The argument `fix_signs` provides two ways to ensure uniqueness of W:

- If argument `fix_signs` is set to `"scores"`, the signs in W are fixed such that the generalized skewness values of all components are positive. If `S1` and `S2` provide location components, which are denoted by T_1 and T_2, the generalized skewness values are computed as $T_1(Z) - T_2(Z)$. Otherwise, the skewness is computed by subtracting the column medians of Z from the corresponding column means so that all components are right-skewed. This way of fixing the signs is preferred in an invariant coordinate selection framework.
- If argument `fix_signs` is set to `"W"`, the signs in W are fixed independently of Z such that the maximum element in each row of W is positive and that each row has norm 1. This is the usual way of fixing the signs in an independent component analysis framework.

In principal, the order of S_1 and S_2 does not matter if both are true scatter matrices. Changing their order will just reverse the order of the components and invert the corresponding generalized kurtosis values.

The same does not hold when at least one of them is a shape matrix rather than a true scatter matrix. In that case, changing their order will also reverse the order of the components, but the ratio of the generalized kurtosis values is no longer 1 but only a constant. This is due to the fact that when shape matrices are used, the generalized kurtosis values are only relative ones.

Different algorithms are available to compute the invariant coordinate system of a data frame X_n with n observations:

- **"whiten"**: whitens the data X_n with respect to the first scatter matrix before computing the second scatter matrix. If `S2` is not a function, whitening is not applicable.
  - whiten the data X_n with respect to the first scatter matrix: $Y_n = X_n S_1(X_n)^{-1/2}$
  - compute S_2 for the uncorrelated data: $S_2(Y_n)$
  - perform the eigendecomposition of $S_2(Y_n)$: $S_2(Y_n) = UDU'$
  - compute W: $W = U' S_1(X_n)^{-1/2}$
- **"standard"**: performs the spectral decomposition of the symmetric matrix $M(X_n)$
  - compute $M(X_n) = S_1(X_n)^{-1/2} S_2(X_n) S_1(X_n)^{-1/2}$
  - perform the eigendecomposition of $M(X_n)$: $M(X_n) = UDU'$
  - compute W: $W = U' S_1(X_n)^{-1/2}$
- **"QR"**: numerically stable algorithm based on the QR algorithm for a common family of scatter pairs: if `S1` is `ICS_cov()` or `cov()`, and if `S2` is one of `ICS_cov4()`, `ICS_covW()` , `ICS_covAxis()`, `cov4()`, `covW()`, or `covAxis()`. For other scatter pairs, the QR algorithm is not applicable. See Archimbaud et al. (2023) for details.

The "whiten" algorithm is the most natural version and therefore the default. The option "standard" should be only used if the scatters provided are not functions but precomputed matrices. The option "QR" is mainly of interest when there are numerical issues when "whiten" is used and the scatter combination allows its usage.

Note that when the purpose of ICS is outlier detection the package ICSOutlier provides additional functionalities as does the package `ICSClust` in case the goal of ICS is dimension reduction prior clustering.

Value

An object of class `"ICS"` with the following components:

`gen_kurtosis`	a numeric vector containing the generalized kurtosis values of the invariant coordinates.
`W`	a numeric matrix in which each row contains the coefficients of the linear transformation to the corresponding invariant coordinate.
`scores`	a numeric matrix in which each column contains the scores of the corresponding invariant coordinate.
`gen_skewness`	a numeric vector containing the (generalized) skewness values of the invariant coordinates (only returned if `fix_signs = "scores"`).
`S1_label`	a character string providing a label for the first scatter matrix to be used by various methods.
`S2_label`	a character string providing a label for the second scatter matrix to be used by various methods.
`S1_args`	a list containing additional arguments used to compute `S1` (if a function was supplied).
`S2_args`	a list containing additional arguments used to compute `S2` (if a function was supplied).
`algorithm`	a character string specifying how the invariant coordinate is computed.

`center`	a logical indicating whether or not the data were centered with respect to the first location vector before computing the invariant coordinates.
`fix_signs`	a character string specifying how the signs of the invariant coordinates were fixed.

Author(s)

Andreas Alfons and Aurore Archimbaud, based on code for `ics()` and `ics2()` by Klaus Nordhausen

References

Tyler, D.E., Critchley, F., Duembgen, L. and Oja, H. (2009) Invariant Co-ordinate Selection. *Journal of the Royal Statistical Society, Series B*, **71**(3), 549–592. doi:10.1111/j.14679868.2009.00706.x.

Archimbaud, A., Drmac, Z., Nordhausen, K., Radojcic, U. and Ruiz-Gazen, A. (2023) Numerical Considerations and a New Implementation for Invariant Coordinate Selection. *SIAM Journal on Mathematics of Data Science*, **5**(1), 97–121. doi:10.1137/22M1498759.

See Also

`gen_kurtosis()`, `coef()`, `components()`, `fitted()`, and `plot()` methods

Examples

```
# import data
data("iris")
X <- iris[,-5]

# run ICS
out_ICS <- ICS(X)
out_ICS
summary(out_ICS)

# extract generalized eigenvalues
gen_kurtosis(out_ICS)
# Plot
screeplot(out_ICS)

# extract the components
components(out_ICS)
components(out_ICS, select = 1:2)

# Plot
plot(out_ICS)

# equivalence with previous functions
out_ics <- ics(X, S1 = cov, S2 = cov4, stdKurt = FALSE)
out_ics
out_ics2 <- ics2(X, S1 = MeanCov, S2 = Mean3Cov4)
out_ics2
out_ICS
```

```
# example using two functions
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:
if (require("ICSNP")) {
  ics.X.2 <- ICS(X, S1 = tyler.shape, S2 = duembgen.shape,
  S1_args = list(location=0))
  summary(ics.X.2)
  plot(ics.X.2)
  # example using three pictures
  library(pixmap)
  fig1 <- read.pnm(system.file("pictures/cat.pgm", package = "ICS")[1],
                   cellres = 1)
  fig2 <- read.pnm(system.file("pictures/road.pgm", package = "ICS")[1],
                   cellres = 1)
  fig3 <- read.pnm(system.file("pictures/sheep.pgm", package = "ICS")[1],
                   cellres = 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)
  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, cellres = 1))
  plot(pixmapGrey(X.mixed[,2], ncol = p, cellres = 1))
  plot(pixmapGrey(X.mixed[,3], ncol = p, cellres = 1))
  plot(pixmapGrey(components(ICA.fig)[,1], ncol = p, cellres = 1))
  plot(pixmapGrey(components(ICA.fig)[,2], ncol = p, cellres = 1))
  plot(pixmapGrey(components(ICA.fig)[,3], ncol = p, cellres = 1))
}
```

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, there 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 which 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`.

Note

Function `ics2()` reached the end of its lifecycle, please use `ICS()` instead. In future versions, `ics2()` will be deprecated and eventually removed.

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`

ICS_scatter *Location and Scatter Estimates for ICS*

Description

Computes a scatter matrix and an optional location vector to be used in transforming the data to an invariant coordinate system or independent components.

Usage

```
ICS_cov(x, location = TRUE)

ICS_cov4(x, location = c("mean", "mean3", "none"))

ICS_covW(x, location = TRUE, alpha = 1, cf = 1)

ICS_covAxis(x, location = TRUE)

ICS_tM(x, location = TRUE, df = 1, ...)

ICS_scovq(x, y, ...)
```

Arguments

x	a numeric matrix or data frame.
location	for ICS_cov(), ICS_cov4(), ICS_covW(), and ICS_covAxis(), a logical indicating whether to include the sample mean as location estimate (default to TRUE). For ICS_cov4(), alternatively a character string specifying the location estimate can be supplied. Possible values are "mean" for the sample mean (the default), "mean3" for a location estimate based on third moments, or "none" to not include a location estimate. For ICS_tM() a logical inficating whether to include the M-estimate of location (default to TRUE).
alpha	parameter of the one-step M-estimator (default to 1).
cf	consistency factor of the one-step M-estimator (default to 1).
df	assumed degrees of freedom of the t-distribution (default to 1, which corresponds to the Cauchy distribution).
...	additional arguments to be passed down to scovq().
y	numerical vector specifying the dependent variable.

Details

ICS_cov() is a wrapper for the sample covariance matrix as computed by cov().

ICS_cov4() is a wrapper for the scatter matrix based on fourth moments as computed by cov4(). Note that the scatter matrix is always computed with respect to the sample mean, even though the returned location component can be specified to be based on third moments as computed by mean3(). Setting a location component other than the sample mean can be used to fix the signs of the invariant coordinates in ICS() based on generalized skewness values, for instance when using the scatter pair ICS_cov() and ICS_cov4().

ICS_covW() is a wrapper for the one-step M-estimator of scatter as computed by covW().

ICS_covAxis() is a wrapper for the one-step Tyler shape matrix as computed by covAxis(), which is can be used to perform Principal Axis Analysis.

ICS_tM() is a wrapper for the M-estimator of location and scatter for a multivariate t-distribution, as computed by tM().

ICS_scovq() is a wrapper for the supervised scatter matrix based on quantiles scatter, as computed by scovq().

Value

An object of class "ICS_scatter" with the following components:

location	if requested, a numeric vector giving the location estimate.
scatter	a numeric matrix giving the estimate of the scatter matrix.
label	a character string providing a label for the scatter matrix.

Author(s)

Andreas Alfons and Aurore Archimbaud

References

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**(12), 2981–3000. doi:10.1080/03610929508831664.

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

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**(2), 441–453. doi:10.1080/03610919408813180.

Oja, H., Sirkia, S. and Eriksson, J. (2006) Scatter Matrices and Independent Component Analysis. *Austrian Journal of Statistics*, **35**(2&3), 175-189.

Tyler, D.E., Critchley, F., Duembgen, L. and Oja, H. (2009) Invariant Co-ordinate Selection. *Journal of the Royal Statistical Society, Series B*, **71**(3), 549–592. doi:10.1111/j.14679868.2009.00706.x.

See Also

ICS()

colMeans(), mean3()

cov(), cov4(), covW(), covAxis(), tM(), scovq()

Examples

```
data("iris")
X <- iris[,-5]
ICS_cov(X)
ICS_cov4(X)
ICS_covW(X, alpha = 1, cf = 1/(ncol(X)+2))
ICS_covAxis(X)
ICS_tM(X)


# 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
ICS_scovq(X, y = y)
```

---

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 a $n \times p$ 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., Sirki?, 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

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 location 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

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 that 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 an `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)
```

---

plot.ICS-S3 *Scatterplot Matrix of Component Scores from the ICS Transformation*

Description

Produces a scatterplot matrix of the component scores of an invariant coordinate system obtained via an ICS transformation.

Usage

```
## S3 method for class 'ICS'
plot(x, select = NULL, index = NULL, ...)
```

Arguments

x	an object inheriting from class "ICS" containing results from an ICS transformation.

`select`	an integer, character, or logical vector specifying which components to plot. If `NULL`, all components are plotted if there are at most six components, otherwise the first three and the last three components are plotted (as the components with extreme generalized kurtosis values are the most interesting ones).
`index`	an integer vector specifying which components to plot, or `NULL` to plot all components. Note that `index` is deprecated and may be removed in the future, use `select` instead.
`...`	additional arguments to be passed down to `pairs()`.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`gen_kurtosis()`, `coef()`, `components()`, and `fitted()` methods

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
plot(out)
plot(out, select = c(1,4))
```

`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.ICS-S3 *Basic information of ICS Object*

Description

Prints information of an ICS object.

Usage

```
## S3 method for class 'ICS'
print(x, info = FALSE, digits = 4L, ...)
```

Arguments

x	object of class ICS.
info	Logical, either TRUE or FALSE. If TRUE, print additional information on arguments used for computing scatter matrices (only named arguments that contain numeric, character, or logical scalars) and information on the parameters of the algorithm. Default is FALSE.
digits	number of digits for the numeric output.
...	additional arguments passed to print()

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

ICS()

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
print(out)
print(out, info = TRUE)
```

`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 otherwise. 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="")
```

---

screeplot.ICS-S3 *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)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`gen_kurtosis()` method

Examples

```
X <- iris[,-5]
out <- ICS(X)
screeplot(out)
screeplot(out, type = "lines")
```

---

summary.ics — *To summarize an 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.ICS-S3` *To summarize an* `ICS` *object*

Description

Summarizes and prints an `ICS` object in an informative way.

Usage

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

Arguments

`object`	object of class `ICS`.
`...`	additional arguments passed to `print.ICS()`.

Author(s)

Andreas Alfons and Aurore Archimbaud

See Also

`ICS()`

`print.ICS()`

Examples

```
data("iris")
X <- iris[,-5]
out <- ICS(X)
summary(out)
```

`summary.ics2` *To summarize an 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

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 function 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