Package 'ICSShiny'

October 12, 2022

Type Package

Title ICS via a Shiny Application

Version 0.5

Date 2018-04-01

Author Aurore Archimbaud, Joris May, Klaus Nordhausen, Anne Ruiz-Gazen

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

Description Performs Invariant Coordinate Selection (ICS) (Tyler, Critchley, Duembgen and Oja (2009) <doi:10.1111/j.1467-9868.2009.00706.x>) and especially ICS for multivariate outlier detection with application to quality control (Archimbaud, Nordhausen, Ruiz-Gazen (2016) <arXiv:1612.06118>) using a shiny app.

License GPL (>= 2)

Depends ICS, ICSOutlier, shiny

Imports ICSNP, rrcov, simsalapar, DT

NeedsCompilation no

Repository CRAN

Date/Publication 2018-04-01 20:21:35 UTC

R topics documented:

HRMEST *Wrapper for Joint Affine Equivariant Estimation of Multivariate Median and Tyler's Shape Matrix*

Description

The function returns, for some multivariate data, the joint affine equivariant estimation of multivariate median and Tyler's shape matrix obtained from HR.Mest.

Usage

```
HRMEST(x, ...)
```

Arguments

x	numeric data matrix or dataframe.
...	further arguments passed on to HR.Mest.

Details

The use of the ICSShiny function requires to pass as an argument a list with: the location vector and the scatter matrix, as the first two arguments.

The HRMEST function is mainly for internal use in the ICSShiny application.

Value

location	the location vector obtained from the joint affine equivariant estimation of multivariate median and Tyler's shape matrix.
scatter	the scatter matrix obtained from the joint affine equivariant estimation of multivariate median and Tyler's shape matrix.

Author(s)

Klaus Nordhausen

References

HR.Mest

See Also

HR.Mest, ICSShiny

Examples

```
library(ICSShiny)
res.HR.Mest <- HRMEST(iris[, 1:4], maxiter = 1000)
```

ICSShiny *Invariant Coordinate Selection With a Shiny App*

Description

Performs ICS via a shiny app where the user can change the scatter matrices, explore the output and download graphs and components. Also the ICS outlier detection framework, from the ICSOutlier package is available. It is inspired from the Factoshiny application of the FactoMineR package.

Usage

```
ICSShiny(x, S1 = MeanCov, S2 = Mean3Cov4,
         S1args = list(), S2args = list(), seed = NULL,
         ncores = NULL, iseed = NULL,
         pkg = "ICSOutlier")
```

Arguments

x	data matrix or dataframe with at least two numeric variables. Please note that it can contain non-numeric variables, but ICS is only performed on numeric variables.
S1	name of the function which returns the first location vector T1 and scatter matrix S1. See details and ics2 for more information. Default is MeanCov.
S2	name of the function which returns the second location vector T2 and scatter matrix S2. See details and ics2 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.
seed	to fix a seed when needed in order to fix the thresholds. Default is NULL. See details for more information.
ncores	number of cores to be used in dist.simu.test and comp.simu.test. If NULL or 1, no parallel computing is used. Otherwise makeCluster with type = "PSOCK" is used.
iseed	If parallel computation is used the seed passed on to clusterSetRNGStream. Default is NULL which means no fixed seed is used.
pkg	When using parallel computing, a character vector listing all the packages which need to be loaded on the different cores via require. Must be at least "ICSOutlier" and must contain the packages needed to compute the scatter matrices.

Details

Choice of the parameters The scatter matrices and their associated location estimators can be selected through the list out of the options: MeanCov, Mean3Cov4, MCD, TM. It is also possible to run the application with your own functions as long as they are passed as an argument of the call to ICSShiny. However, in this case it is not possible to run the simulations' steps for now. ICS is only performed on numeric variables. Only non-numeric variables are proposed for labelling and/or categorizing the observations.

Component selection For computing the kernel densities in the second sub-tab, the weight is given by the Gaussian function and the bandwidth follows the rule of thumb of Silverman (1986).

For the automatic selection of the Invariant Components (IC), the referenced normality tests are the same as in the `comp.norm.test` function: `"jarque.test"`, `"anscombe.test"`, `"bonett.test"`, `"agostino.test"`, `"shapiro.test"`. All the decisions are corrected from multiple testing by adjusting the levels as in `comp.norm.test`. The number of components to keep can also be decided from Monte Carlo simulations trough the `comp.simu.test` function. This parallel analysis method may need a very long time to compute, so it is used only if the user clicks on the 'Launch the test' button.

Value

Returns several tabs on the navigator:

`Choice of the parameters`
: The scatterplot matrix of an ICS object for the parameters chosen on the left part (variables included/excluded, the location vectors and scatter matrices).

`Component selection`
: Three different subtabs to help the user to choose the interesting components. The first sub-tab is the screeplot of the eigenvalues of the ICS object followed by the summary of the analysis. The second sub-tab plots the kernel density of the ICS components. The third sub-tab suggests which components to select, starting from the highest and/or the lowest kurtosis, through different normality tests or simulations.

  The default values of the slidebar in the left are obtained from `"agostino.test"` at 5%.

`Matrix scatterplot of invariant components`
: The two sub-tabs aim at identifying groups or outliers by using pairwise plots of invariant coordinates. It offers two ways of plotting them: only two invariant components or a scatterplot matrix with up to six invariant components. The left panel allows to color the groups identified by the user and label the observations.

`Outlier identification`
: This tab plots outlyingness values for each observation based on the selected components. These squared ICS distances are computed through the `ics.distances` function as the Euclidian distance of the observations to the origin using the selected centered components. The identification of the outliers can be based on different cut-offs: from Monte Carlo simulations as in `dist.simu.test` or by giving a percentage or a number of observations to identify.

`Descriptive statistics`
: This tab gives some descriptive statistics on different subsets of the data (for all the observations, for the observations from a given cluster, for the outlying observations) and enables to compare the sub-populations. The application includes a boxplot, a kernel density, an histogram and some basic statistics: Min, Q1, Mean, Median, Q3 and Max.

`Data Table`
: This tab contains the dataset with a nice display and the possibility to choose different sub-populations of the data: all the observations, the observations from a given cluster or the outlying observations.

`Save` This tab allows to display and save the data table of components and the summary of operations. The data frame contains the components kept in the analysis as well as the distance generated by these components. It also includes the cluster the observation belongs to whether the observation is defined as an outlier, as well as the variables used for labelling and categorizing the data. The data are saved in a csv format. The summary of operations contains a summary of all parameters that were used to obtain the current result, it may be useful for another user who may want to get the same result as the original user. It is saved in a txt format.

The "Close the session" button closes the application and saves the icsshiny object into the global environment.

Author(s)

Aurore Archimbaud and Joris May

References

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., Nordhausen, K. and Ruiz-Gazen, A. (2016), ICS for multivariate outlier detection with application to quality control, <https://arxiv.org/pdf/1612.06118.pdf>.

See Also

`ics2`,`ics.outlier`,
shiny website

Examples

```
if(interactive()){
library(ICSShiny)
# ICS with ICSShiny:
res.shiny <- ICSShiny(iris)

# Close the session by clicking on the button or closing the navigator's tab
# ICS on a result of an ICSshiny object
ICSShiny(res.shiny)

# ICS with ICSShiny and different parameters
res.shiny <- ICSShiny(iris, S1 = MCD, S1args=list(alpha=0.7), seed = 7587)

# ICS with ICSShiny with parallelization of computations and seed
res.shiny <- ICSShiny(iris, iseed = 1234, ncores = 2)
}
```

MCD *Wrapper for MCD location and scatter estimates*

Description

The function returns, for some multivariate data, the MCD location and scatter estimates obtained from CovMcd.

Usage

```
MCD(x, ...)
```

Arguments

x	numeric data matrix or dataframe.
...	further arguments passed to or from other methods.

Details

The use of the ICSShiny function requires to pass as an argument a list with: the location vector and the scatter matrix, as the first two arguments.

The MCD function is proposed inside the ICSShiny application.

Value

location	MCD location vector.
scatter	MCD scatter estimate.

Author(s)

Aurore Archimbaud and Joris May

References

CovMcd

See Also

CovMcd, ICSShiny

Examples

```
library(ICSShiny)
res.MCD <- MCD(iris[, 1:4], alpha = 0.75)
```

`print.icsshiny` *Prints the ICSshiny Results*

Description

Prints an object of class icsshiny, typically the results of a call to `ICSShiny`. The output corresponds to the summary of operations.

Usage

```
## S3 method for class 'icsshiny'
print(x, ...)
```

Arguments

`x`	an object of class icsshiny.
`...`	further arguments to be passed to or from methods.

Author(s)

Aurore Archimbaud and Joris May

See Also

`ICSShiny`

Examples

```
## Not run:
library(ICSShiny)
# ICS with Factoshiny:
res.shiny <- ICSshiny(iris)

# click on the "Close the session" button or close the tab
print(res.shiny)


## End(Not run)
```

TM *Wrapper for Joint M-estimation of location and scatter for a multivariate t-distribution*

Description

The function returns, for some multivariate data, the joint M-estimation of location and scatter matrix for a multivariate t-distribution obtained from tM.

Usage

```
TM(x, ...)
```

Arguments

x	numeric data matrix or dataframe.
...	further arguments passed to or from other methods.

Details

The use of the ICSShiny function requires to pass as an argument a list with: the location vector and the scatter matrix, as the first two arguments.

The TM function is proposed inside the ICSShiny application.

Value

location	the location vector obtained from the joint M-estimation of location and scatter for a multivariate t-distribution.
scatter	the scatter matrix obtained from the joint M-estimation of location and scatter for a multivariate t-distribution.

Author(s)

Aurore Archimbaud and Joris May

References

tM

See Also

tM, ICSShiny

Examples

```
library(ICSShiny)
res.TM <- TM(iris[, 1:4], df=3)
```

Index