Package: fsdaR (via r-universe)

October 10, 2026

Title Robust Data Analysis Through Monitoring and Dynamic Visualization

Version 0.9-1

VersionNote Released 0.9-0 on 2023-12-06 on CRAN

Description Provides interface to the 'MATLAB' toolbox 'Flexible Statistical Data Analysis (FSDA)' which is comprehensive and computationally efficient software package for robust statistics in regression, multivariate and categorical data analysis. The current R version implements tools for regression: (forward search, S- and MM-estimation, least trimmed squares (LTS) and least median of squares (LMS)), for multivariate analysis (forward search, S- and MM-estimation), for cluster analysis and cluster-wise regression. The distinctive feature of our package is the possibility of monitoring the statistics of interest as a function of breakdown point, efficiency or subset size, depending on the estimator. This is accompanied by a rich set of graphical features, such as dynamic brushing, linking, particularly useful for exploratory data analysis.

Depends R (>= 3.5.0)

Imports rJava, methods, stats4, ggplot2

Suggests robustbase, rrcov, MASS

Encoding UTF-8

SystemRequirements (license-free) MATLAB Runtime (MCR) V 9.12, Java (>=8)

LazyLoad yes

LazyData yes

License GPL (>= 3)

URL https://github.com/UniprJRC/fsdaR

BugReports https://github.com/UniprJRC/fsdaR/issues

Author Valentin Todorov [aut, cre] (ORCID:
<https://orcid.org/0000-0003-4215-0245>), Emmanuele Sordini
[aut], Aldo Corbellini [ctb], Francesca Torti [ctb], Marco
Riani [ctb], Domenico Perrotta [ctb], Andrea Cerioli [ctb]

Maintainer Valentin Todorov <valentin.todorov@chello.at>

NeedsCompilation no

RoxygenNote 7.3.3

Config/pak/sysreqs default-jdk

Repository https://cran.r-universe.dev

Date/Publication 2026-05-19 23:35:40 UTC

RemoteUrl https://github.com/cran/fsdaR

RemoteRef HEAD

RemoteSha 405df763a4424a83e46344e61dc1b3c869946cb8

Contents

bank_data . . . 4
carbikeplot . . . 5
corfwdplot . . . 6
covplot . . . 8
diabetes . . . 11
emilia2001 . . . 12
fanplot . . . 13
fishery . . . 16
fishery2003 . . . 17
flea . . . 17
forbes . . . 18
fsdalms.object . . . 19
fsdalts.object . . . 20
fsmeda.object . . . 21
fsmmmdrs . . . 22
fsmmmdrs.object . . . 25
fsmult . . . 26
fsmult.object . . . 30
fsr.object . . . 31
FSR_control . . . 32
fsrbase . . . 35
fsreda.object . . . 38
FSReda_control . . . 40
fsreg . . . 41
fsrfan . . . 44
fsrfan.object . . . 51
geyser2 . . . 52
hawkins . . . 53
hospital . . . 53

Income1 . . . 54
Income2 . . . 55
levfwdplot . . . 55
loyalty . . . 60
LXS_control . . . 61
M5data . . . 63
malfwdplot . . . 64
malindexplot . . . 69
mdrplot . . . 70
mmdplot . . . 74
mmdrsplot . . . 77
mmmult . . . 80
mmmult.object . . . 82
mmmulteda.object . . . 83
mmreg.object . . . 84
MMreg_control . . . 85
mmregeda.object . . . 87
MMregeda_control . . . 88
multiple_regression . . . 90
mussels . . . 91
myrng . . . 91
normBoxCox . . . 92
normYJ . . . 95
normYJpn . . . 97
poison . . . 99
psifun . . . 99
regspmplot . . . 102
resfwdplot . . . 106
resindexplot . . . 111
score . . . 114
score.object . . . 117
scoreYJ . . . 117
scoreYJ.object . . . 119
scoreYJpn . . . 120
scoreYJpn.object . . . 122
smult . . . 123
smult.object . . . 126
smulteda.object . . . 127
spmplot . . . 128
sreg.object . . . 131
Sreg_control . . . 132
sregeda.object . . . 134
Sregeda_control . . . 135
summary.fsdalms . . . 137
summary.fsdalts . . . 138
summary.fsr . . . 140
swissbanknotes . . . 141
swissheads . . . 142

tclusteda.object . . . 143
tclustfsda . . . 144
tclustfsda.object . . . 152
tclustIC . . . 154
tclustic.object . . . 158
tclustICplot . . . 159
tclustICsol . . . 161
tclusticsol.object . . . 163
tclustreg . . . 165
tclustreg.object . . . 170
tclustregIC . . . 172
wool . . . 177
X . . . 177
z1 . . . 178

Index **179**

`bank_data` *Bank data (Riani et al., 2014).*

Description

There are 60 observations on a response y with the values of three explanatory variables. The scatter plot matrix of the data shows y increasing with each of x1, x2 and x3. The plot of residuals against fitted values shows no obvious pattern. However the FS finds that there are 6 masked outliers.

Usage

```
data(bank_data)
```

Format

A data frame with 1949 rows and 14 variables. The variables are as follows:

- x1: Personal loans
- x2: Financing and hire-purchase
- x3: Mortgages
- x4: Life insurance
- x5: Share amount
- x6: Bond account
- x7: Current account
- x8: Salary deposits
- x9: Debit cards
- x10: Credit cards
- x11: Telephone banking

- x12: Domestic direct debits
- x13: Money transfers
- y: Profit/loss

Source

Riani, M., Cerioli, A., Atkinson, A. C., and Perrotta, D. (2014). Supplement to "Monitoring robust regression". doi:10.1214/14-EJS897SUPP.

References

Riani, M., Cerioli, A., Atkinson, A. C., and Perrotta, D. (2014). Monitoring robust regression. *Electronic Journal of Statistics*, 8, 642-673.

`carbikeplot` *Produces the carbike plot to find best relevant clustering solutions obtained by* `tclustICsol`

Description

Takes as input the output of function `tclustICsol` (that is a structure containing the best relevant solutions) and produces the car-bike plot. This plot provides a concise summary of the best relevant solutions. This plot shows on the horizontal axis the value of `c` and on the vertical axis the value of `k`. For each solution we draw a rectangle for the interval of values for which the solution is best and stable and a horizontal line which departs from the rectangle for the values of `c` in which the solution is only stable. Finally, for the best value of `c` associated to the solution, we show a circle with two numbers, the first number indicates the ranked solution among those which are not spurious and the second one the ranked number including the spurious solutions. This plot has been baptized 'car-bike', because the first best solutions (in general 2 or 3) are generally best and stable for a large number of values of `c` and therefore will have large rectangles. In addition, these solutions are likely to be stable for additional values of `c` and therefore are likely to have horizontal lines departing from the rectangles (from here the name 'cars'). Finally, local minor solutions (which are associated with particular values of `c` and `k`) do not generally present rectangles or lines and are shown with circles (from here the name 'bikes').

Usage

```
carbikeplot(out, SpuriousSolutions = FALSE, trace = FALSE, ...)
```

Arguments

`out`	An S3 object of class `tclusticsol.object`, (output of `tclustICsol`) containing the relevant solutions.
`SpuriousSolutions`	Wheather to include or not spurious solutions. By default spurios solutions are not included into the plot.
`trace`	Whether to print intermediate results. Default is `trace=FALSE`.
`...`	potential further arguments passed to lower level functions.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Cerioli, A., Garcia-Escudero, L.A., Mayo-Iscar, A. and Riani M. (2017). Finding the Number of Groups in Model-Based Clustering via Constrained Likelihoods, *Journal of Computational and Graphical Statistics*, pp. 404-416, https://doi.org/10.1080/10618600.2017.1390469.

Examples

```
## Not run:

##  Car-bike plot for the geyser data ========================

data(geyser2)
out <- tclustIC(geyser2, whichIC="MIXMIX", plot=FALSE, alpha=0.1)

## Find the best solutions using as Information criterion MIXMIX
print("Best solutions using MIXMIX")
outMIXMIX <- tclustICsol(out, whichIC="MIXMIX", plot=FALSE, NumberOfBestSolutions=6)

print(outMIXMIX$MIXMIXbs)

carbikeplot(outMIXMIX)

##  Car-bike plot for the flea data ===========================

data(flea)
Y <- as.matrix(flea[, 1:(ncol(flea)-1)])    # select only the numeric variables
rownames(Y) <- 1:nrow(Y)
head(Y)

out <- tclustIC(Y, whichIC="CLACLA", plot=FALSE, alpha=0.1, nsamp=100)

##  Find the best solutions using as Information criterion CLACLA
print("Best solutions using CLACLA")
outCLACLA <- tclustICsol(out,whichIC="CLACLA", plot=FALSE, NumberOfBestSolutions=66)
##  Produce the car-bike plot
carbikeplot(outCLACLA)


## End(Not run)
```

`corfwdplot` *Monitoring the correlations between consecutive distances or residuals*

Description

Provides a method for obtaining the maximum empirical efficiency (in case of MM estimates) or maximum empirical breakdownplot (in case of S estimates) or maximum subset size (in case of forward search), using various measures of correlation between the n Mahalanobis distances or residuals at adjacent values of efficiecy, breakdown point or subset size.

Usage

```
corfwdplot(out, trace = FALSE, ...)
```

Arguments

out	An object of S3 class returned by one of the estimation functions with the monitoring option selected (monitoring=TRUE): fsreda.object, sregeda.object, mmregeda.object, fsmeda.object, smulteda.object or mmmulteda.object. This is a list containing the monitoring of minimum Mahalanobis distance in case of multivariate analysis or the monitoring of residuals in case of regression. The needed elements of out are 1. MAL: matrix containing the squared Mahalanobis distances monitored in each step of the forward search. Every row is associated with a unit (row of data matrix Y). This matrix can be created using one of the functions fsmult, smult or mmmult with the monitoring option selected, i.e. monitoring=TRUE. 2. RES: matrix containing the residuals monitored in each step of the forward search. Every row is associated with a unit (row of data matrix Y). This matrix can be created using the function fsreg with the monitoring option selected, i.e. monitoring=TRUE . 3. bdp: a vector containing breakdown point values that have been used, in case of S estimates. 4. eff: a vector containing efficiency values that have been used, in case of MM estimates.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

A ggplot plot object which can be printed on screen or to a file.

Author(s)

FSDA team, <valentin.todorov@chello.at>

Examples

```
## Not run:

data(hbk, package="robustbase")
(out <- fsmult(hbk[,1:3], monitoring=TRUE))
corfwdplot(out)
```

```
(out1 <- smult(hbk, monitoring=TRUE, trace=TRUE))
corfwdplot(out1)

(out2 <- mmmult(hbk[,1:3], monitoring=TRUE, trace=TRUE))
corfwdplot(out2)

(out3 <- fsreg(hbk[,1:3], hbk[,4], monitoring=TRUE, trace=TRUE, method="FS"))
corfwdplot(out3)

(out4 <- fsreg(hbk[,1:3], hbk[,4], monitoring=TRUE, trace=TRUE, method="S"))
corfwdplot(out4)

(out5 <- fsreg(hbk[,1:3], hbk[,4], monitoring=TRUE, trace=TRUE, method="MM"))
corfwdplot(out5)


## End(Not run)
```

---

covplot *Monitoring of the covariance matrix*

Description

Plots the trajectories of the elements of the covariance (correlation) matrix monitored

Usage

```
covplot(
  out,
  xlim,
  ylim,
  xlab,
  ylab,
  main,
  lwd,
  lty,
  col,
  cex.lab,
  cex.axis,
  subsize,
  fg.thresh,
  fg.unit,
  fg.labstep,
  fg.lwd,
  fg.lty,
  fg.col,
```

```
  fg.mark,
  fg.cex,
  standard,
  fground,
  tag,
  datatooltip,
  trace = FALSE,
  ...
)
```

Arguments

out	An object of S3 class fsmeda.object returned by fsmult with monitoring=TRUE - a list containing the monitoring of minimum Mahalanobis distance. The needed elements of out are 1. S2cov: matrix containing the monitoring of the elements of the covariance matrix in each step of the forward search: 2. Un: matrix containing the order of entry of each unit (necessary if datatooltip or databrush is selected). 3. X: The data matrix.
xlim	Controls the x scale in the plot. xlim is a vector with two elements controlling minimum and maximum on the x-axis. Default is to use automatic scale.
ylim	Controls the y scale in the plot. ylim is a vector with two elements controlling minimum and maximum on the y-axis. Default is to use automatic scale.
xlab	A title for the x axis
ylab	A title for the y axis
main	An overall title for the plot
lwd	The line width, a positive number, defaulting to 1
lty	The line type. Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.
col	Colors to be used for the highlighted units
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
subsize	Numeric vector containing the subset size with length equal to the number of columns of matrix of mahalanobis distances. The default value of subsize is (nrow(MAL) - ncol(MAL) + 1):nrow(MAL)
fg.thresh	(alternative to fg.unit) numeric vector of length 1 or 2 which specifies the highlighted trajectories. If length(fg.thresh) == 1 the highlighted trajectories are those of units that throughtout the search had at leat once a mahalanobis distance greater than fg.thresh. The default value is fg.thresh=2.5. If

length(fg.thresh) == 2 the highlighted trajectories are those of units that throughtout the search had a mahalanobis distance at least once bigger than fg.thresh[2] or smaller than fg.thresh[1].

fg.unit (alternative to fg.thresh), vector containing the list of the units to be highlighted. If fg.unit is supplied, fg.thresh is ignored.

fg.labstep numeric vector which specifies the steps of the search where to put labels for the highlighted trajectories (units). The default is to put the labels at the initial and final steps of the search. fg.labstep='' means no label.

fg.lwd The line width for the highlighted trajectories (units). Default is 1.

fg.lty The line type for the highlighted trajectories (units). Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.

fg.col colors to be used for the highlighted units.

fg.mark Controlls whether to plot highlighted trajectories as symbols. if fg.mark==TRUE each line is plotted using a different symbol else no marker is used (default).

fg.cex Controls the font size of the labels of the trajectories in foreground. If fg.cex=0 no labels will be shown - equivalent to fg.labstop="".

standard MATLAB-style arguments - appearance of the plot in terms of xlim, ylim, axes labels and their font size style, color of the lines, etc.

fground MATLAB-style arguments - for the trajectories in foregroud.

tag Plot handle. String which identifies the handle of the plot which is about to be created. The default is tag='pl_mmd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

datatooltip If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function datacursormode() for more details or see the examples below). The default options are DisplayStyle="Window" and SnapToDataVertex="on".

trace Whether to print intermediate results. Default is trace=FALSE.

... potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

Examples

```
## Not run:
X <- iris[,1:4]
out <- fsmult(X, monitoring=TRUE)

##  Produce monitoring covariances plot with all the default options
covplot(out)


## End(Not run)
```

`diabetes` *Diabetes data*

Description

The diabetes dataset, introduced by Reaven and Miller (1979), consists of 145 observations (patients). For each patient three measurements are reported: plasma glucose response to oral glucose, plasma insulin response to oral glucose, degree of insulin resistance.

Usage

```
data("diabetes")
```

Format

A data frame with the following variables:

glucose Area under plasma glucose curve after a three hour oral glucose tolerance test (OGTT).

insulin Area under plasma insulin curve after a three hour oral glucose tolerance test (OGTT).

sspg Steady state plasma glucose.

class The type of diabete: `Normal`, `Overt`, and `Chemical`.

Source

Reaven, G. M. and Miller, R. G. (1979). An attempt to define the nature of chemical diabetes using a multidimensional analysis. *Diabetologia* 16:17-24.

Examples

```
library(rrcov)
data(diabetes)
head(diabetes)
plot(CovMcd(diabetes[, 1:3]), which="pairs", col=diabetes$class)
```

`emilia2001` *Demographic data from the 341 miniciplaities in Emilia Romagna (an Italian region).*

Description

A data set containing 28 demographic variables for 341 municipalities in Emilia Romagna (an Italian region).

Usage

```
data(emilia2001)
```

Format

A data frame with 341 rows and 28 variables The variables are as follows:

- less10: population aged less than 10
- more75: population aged more than 75
- single single-member families
- divorced": divorsed
- widows: widows and widowers
- graduates: population aged more than 25 who are graduates
- no_education: of those aged over 6 having no education
- employed: activity rate
- unemployed: unemployment rate
- increase_popul: standardised natural increase in population
- migration: standardised change in population due to migration
- birth_92_94: average birth rate over 1992-94
- fecundity: three-year average birth rate amongst women of child-bearing age
- houses: occupied houses built since 1982
- houses_2WCs: occupied houses with 2 or more WCs
- houses_heating: occupied houses with fixed heating system
- TV: TV licence holders
- cars: number of cars for 100 inhabitants
- luxury_cars: luxury cars
- hotels: working in hotels and restaurants
- banking: working in banking and finance
- income: average declared income amongst those filing income tax returns
- income_tax_returns: inhabitants filing income tax returns

- factories: residents employed in factories and public services
- factories_more10: employees employed in factories withy more tha 10 employees
- factories_more50: employees employed in factories withy more tha 50 employees
- artisanal: artisanal enterprises
- entrepreneurs: enterpreneous and skilled self-employed among those of working age

@references Atkinson, A. C., Riani, M., and Cerioli, A. (2004). *Exploring Multivariate Data with the Forward Search*. Springer-Verlag, New York.

`fanplot` *Plots the fan plot for transformation in linear regression*

Description

Plots the fan plot for transformation in linear regression

Usage

```
fanplot(
  out,
  xlab = "Subset of size m",
  ylab = "Score test statistic",
  main = "Fan plot",
  xlim,
  ylim,
  cex.lab,
  cex.axis,
  lwd = 2,
  lwd.env = 1,
  tag,
  datatooltip,
  label,
  nameX,
  databrush,
  conflev,
  trace = FALSE,
  ...
)
```

Arguments

`out`	An object of S3 class `fsrfan.object` returned by `fsrfan` - a list containing the following elements: • `Score`: matrix of size `(n-init)` x `length(la)+1` with the fwd search index in the first column and the value of the score test in each step of the fwd search for `la[i]` in each of the following `i` columns.

- la: vector containing the values of lambda for which fan plot is constructed.
- bs: matrix of size p x length(la) containing the units forming the initial subset for each value of lambda.
- Un: a three-dimensional array containing length(la) matrices of size retnUn=(n-init) X retpUn=11. Each matrix contains the unit(s) included in the subset at each step in the search associated with the corresponding element of la.
- y: a vector containing the response (necessary only if option databrush is set).
- X: a matrix containing the explanatory variables (necessary only if option databrush is set).

xlab — A title for the x axis

ylab — A title for the y axis

main — An overall title for the plot

xlim — Controls the x scale in the plot. xlim is a vector with two elements controlling minimum and maximum on the x-axis. Default is to use automatic scale.

ylim — Controls the y scale in the plot. ylim is a vector with two elements controlling minimum and maximum on the y-axis. Default is to use automatic scale.

cex.lab — The magnification to be used for x and y labels relative to the current setting of cex

cex.axis — The magnification to be used for axis annotation relative to the current setting of cex

lwd — The line width, a positive number, default is lwd=1

lwd.env — Width of the envelope lines. Default is lwd.env=1.

tag — Plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_resfwd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

datatooltip — Interactive clicking. It is inactive if this parameter is set to FALSE. The default is datatooltip=TRUE, the user can select with the mouse an individual mahalanobis distance trajectory in order to have information about the corresponding unit, the associated label and the step of the search in which the unit enters the subset. If datatooltip is a list it may contain the following fields:

1. DisplayStyle determines how the data cursor displays. Possible values are 'datatip' and 'window' (default). 'datatip' displays data cursor information in a small yellow text box attached to a black square marker at a data point you interactively select. 'window' displays data cursor information for the data point you interactively select in a floating window within the figure.
2. SnapToDataVertex: specifies whether the data cursor snaps to the nearest data value or is located at the actual pointer position. Possible values are SnapToDataVertex='on' (default) and SnapToDataVertex='off'.

3. LineColor: controls the color of the trajectory selected with the mouse. It can be an RGB triplet of values between 0 and 1, or character vector indicating a color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation.
4. SubsetLinesColor: enables to control the color of the trajectories of the units that are in the subset at a given step of the search (if `resfwdplot()` is applied to an object of class `fsreda.object`) or have a weight greater than 0.9 (if `resfwdplot()` is applied to an object of class `sregeda.object` or `mmregeda.object`). This can be done (repeatedly) with a left mouse click in proximity of the step of interest. A right mouse click will terminate the selection by marking with a up-arrow the step corresponding to the highlighted lines. The highlighted lines by default are in red, but a different color can be specified as RGB triplet or character of color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation. By default `SubsetLinesColor=""`, i.e. the modality is not active. Any initialization for `SubsetLinesColor` which cannot be interpreted as RGB vector will be converted to blue, i.e. `SubsetLinesColor` will be forced to be [0 0 1]. If `SubsetLinesColor` is not empty the previous option `LineColor` is ignored.

`label` Character vector containing the labels of the units (optional argument used when `datatooltip=TRUE`. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

`nameX` Add variable labels in plot. A vector of strings of length `p` containing the labels of the variables in the dataset. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

`databrush` Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is `TRUE` or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. In addition, brushed units can be highlighted in the monitoring mahalanobis distance plot. Note that the window style of the other figures is set equal to that which contains the monitoring mahalanobis distance plot. In other words, if the monitoring mahalanobis distance plot is docked all the other figures will be docked too.

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist=").

If `databrush=list(...)`, it is possible to use all optional arguments of the MATLAB function `selectdataFS()` and the following optional arguments:

- persist. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
- `label`: add labels of brushed units in the monitoring plot.

- `labeladd`: add labels of brushed units in the scatterplot matrix. If this option is '1', we label the units of the last selected group with the unit row index in the matrix X. The default value is labeladd='', i.e. no label is added.

`conflev` confidence interval for the horizontal bands. It can be a vector of different confidence level values, e.g. c(0.95, 0.99, 0.999). The confidence interval is based on the chi^2 distribution.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C. and Riani, M. (2000), *Robust Diagnostic Regression Analysis* Springer Verlag, New York.

Atkinson, A.C. and Riani, M. (2002), Tests in the fan plot for robust, diagnostic transformations in regression, *Chemometrics and Intelligent Laboratory Systems*, **60**, pp. 87–100.

Atkinson, A.C. Riani, M. and Corbellini A. (2019), The analysis of transformations for profit-and-loss data, *Journal of the Royal Statistical Society, Series C, "Applied Statistics"*, **69**, pp. 251–275. doi:10.1111/rssc.12389

Atkinson, A.C. Riani, M. and Corbellini A. (2021), The Box-Cox Transformation: Review and Extensions, *Statistical Science*, **36**(2), pp. 239–255. doi:10.1214/20STS778.

`fishery` *Fishery data.*

Description

The fishery data consist of 677 transactions of a fishery product in Europe. For each transaction the Value in 1000 euro and the quantity in Tons are reported.

Usage

```
data(fishery)
```

Format

A data frame with 677 rows and 2 variables

fishery2003 *Fishery 2003 data.*

Description

The fishery2003 data consist of 167 transactions of a fishery product in Europe. For each transaction the Value in 1000 euro and the quantity in tones are reported. Similar to the `fishery` data but it contains information on the importing EU country (declarant) and on the period (date) in which the transaction took place. It referes to year 2003.

Usage

```
data(fishery2003)
```

Format

A data frame with 167 rows and 4 variables

flea *Flea*

Description

Flea-beetle measurements

Usage

```
data(flea)
```

Format

A data frame with 74 rows and 7 variables: six explanatory and one response variable - `species`. The variables are as follows:

- tars1: width of the first joint of the first tarsus in microns (the sum of measurements for both tarsi)
- tars2: the same for the second joint
- head: the maximal width of the head between the external edges of the eyes in 0.01 mm
- ade1: the maximal width of the aedeagus in the fore-part in microns
- ade2: the front angle of the aedeagus (1 unit = 7.5 degrees)
- ade3: the aedeagus width from the side in microns
- species, which species is being examined - `Concinna`, `Heptapotamica`, `Heikertingeri`

References

A. A. Lubischew (1962), On the Use of Discriminant Functions in Taxonomy, *Biometrics*, **18**4 pp.455–477.

Examples

```
data(flea)
head(flea)
```

`forbes` *Forbes' data on air pressure in the Alps and the boiling point of water (Weisberg, 1985).*

Description

A data set on air pressure in the Alps and the boiling point of water (Weisberg, 1985). There are 17 observations on the boiling point of water at different pressures, obtained from measurements at a variety of elevations in the Alps. The purpose of the experiment was to allow prediction of pressure from boiling point, which is easily measured, and so to provide an estimate of altitude: the higher the altitude, the lower the pressure. The dataset is characterized by one clear outlier.

Usage

```
data(forbes)
```

Format

A data frame with 17 rows and 2 variables The variables are as follows:

- x: boiling point
- y: 100 x log(pressure)

References

Weisberg, S. (1985). *Applied Linear Regression*. Wiley, New York.

Examples

```
data(forbes)
plot(y~x, data=forbes)
```

fsdalms.object *Description of* fsdalms *Objects*

Description

An object of class fsdalms.object holds information about the result of a call to fsreg.

Value

The object itself is basically a list with the following components:

rew	If rew=TRUE all subsequent output refers to reweighted else no reweighting is done.
beta	p-by-1 vector containing the estimated regression parameters.
bs	p x 1 vector containing the units forming subset associated with bLMS (bLTS).
residuals	residuals.
scale	scale estimate of the residuals.
weights	Vector like y containing weights. The elements of this vector are 0 or 1. These weights identify the h observations which are used to compute the final LTS (LMS) estimate. sum(weights)=h if there is not a perfect fit otherwise sum(weights) can be greater than h
h	The number of observations that have determined the LTS (LMS) estimator, i.e. the value of h.
outliers	vector containing the list of the units declared as outliers using confidence level specified in input scalar conflev.
conflev	confidence level which is used to declare outliers. Remark: conflev will be used to draw the horizontal lines (confidence bands) in the plots. Default value is 0.975
singsub	Number of subsets wihtout full rank. Notice that if this number is greater than 0.1*(number of subsamples) a warning is produced
X	the data matrix X
y	the response vector y

The object has class "fsdalms".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="LMS"))
    class(out)
    summary(out)

## End(Not run)
```

fsdalts.object *Description of* fsdalts *Objects*

Description

An object of class fsdalts.object holds information about the result of a call to fsreg.

Value

The object itself is basically a list with the following components:

rew	If rew=TRUE all subsequent output refers to reweighted else no reweighting is done.
beta	p-by-1 vector containing the estimated regression parameters.
bs	p x 1 vector containing the units forming subset associated with bLMS (bLTS).
residuals	residuals.
scale	scale estimate of the residuals.
weights	Vector like y containing weights. The elements of this vector are 0 or 1. These weights identify the h observations which are used to compute the final LTS (LMS) estimate. sum(weights)=h if there is not a perfect fit otherwise sum(weights) can be greater than h
h	The number of observations that have determined the LTS (LMS) estimator, i.e. the value of h.
outliers	vector containing the list of the units declared as outliers using confidence level specified in input scalar conflev.
conflev	confidence level which is used to declare outliers. Remark: conflev will be used to draw the horizontal lines (confidence bands) in the plots. Default value is 0.975
singsub	Number of subsets wihtout full rank. Notice that if this number is greater than 0.1*(number of subsamples) a warning is produced
X	the data matrix X
y	the response vector y

The object has class "fsdalts".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="LTS"))
    class(out)
    summary(out)

## End(Not run)
```

fsmeda.object *Description of* fsmeda.object *Objects*

Description

An object of class fsmeda.object holds information about the result of a call to fsmult when called with parameter monitoring=TRUE.

Value

The object itself is basically a list with the following components:

MAL: n x (n-init+1) matrix containing the monitoring of Each row represents the distance Mahalanobis distance for the corresponding unit.

BB: n x (n-init+1) matrix containing the information about the units belonging to the subset at each step of the forward search. The first column contains the indexes of the units forming subset in the initial step and each further column - the indexes of the units forming the corresponding step. The last column contains the units forming subset in the final step (all units).

md: n-by-1 vector containing the estimates of the robust Mahalanobis distances (in squared units). This vector contains the distances of each observation from the location of the data, relative to the scatter matrix cov.

mmd: (n-init) x 3 matrix. which contains the monitoring of minimum MD or (m+1)th ordered MD at each step of the forward search.

- 1st column = fwd search index (from init to n-1)
- 2nd column = minimum MD
- 3rd column = (m+1)th-ordered MD

msr: (n-init+1) x 3 matrix which contains the monitoring of maximum MD or m-th ordered MD at each step of the forward search.

- 1st column = fwd search index (from init to n)
- 2nd column = maximum MD
- 3rd column = mth-ordered MD

gap: (n-init+1) x 3 matrix which contains the monitoring of the gap (difference between minMD outside subset and max inside).

- 1st column = fwd search index (from init to n)
- 2nd column = min MD - max MD
- 3rd column = (m+1)th-ordered MD - mth ordered distance

Loc: (n-init+1) x (p+1) matrix which contains the monitoring of the estimated means at each step of the fwd search.

S2cov: (n-init+1) x (p*(p+1)/2+1) matrix which contains the monitoring of the of the elements of the covariance matrix in each step of the forward search.

- 1st column = fwd search index (from init to n)
- 2nd column = monitoring of S[1,1]
- 3rd column = monitoring of S[1,2]
- ...
- last column = monitoring of S[p,p]

`detS`: (n-init+1) x 2 matrix which contains the monitoring of the determinant of the covariance matrix in each step of the forward search.

`Un`: (n-init)-by-11 matrix which contains the unit(s) included in the subset at each step of the fwd search. REMARK: in every step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one. Un[1 ,2] for example contains the unit included in step init+1. Un[end, 2] contains the units included in the final step of the search.

`X`: the data matrix X.

The object has class `"fsmeda"`.

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsmult(hbk[,1:3], monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

`fsmmmdrs` *Performs random start monitoring of minimum Mahalanobis distance*

Description

The trajectories originate from many different random initial subsets and provide information on the presence of groups in the data. Groups are investigated by monitoring the minimum Mahalanobis distance outside the forward search subset.

Usage

```
fsmmmdrs(
  x,
  plot = FALSE,
  init,
  bsbsteps,
  nsimul = 200,
  nocheck = FALSE,
  numpool,
  cleanpool = FALSE,
  msg = FALSE,
```

```
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
plot	Plots the random starts minimum Mahalanobis distance with 1 If `plot=FALSE` (default) or `plot=0` no plot is produced. The scale (ylim) for the y axis is defined as follows: • ylim[2] is the maximum between the values of `mmd` in steps `[n*0.2 n]` and the final value of the 99 per cent envelope multiplied by 1.1. • ylim[1] is the minimum between the values of `mmd` in steps `[n*0.2 n]` and the 1 per cent envelope multiplied by 0.9. Remark: the plot which is produced is very simple. In order to control a series of options in this plot (including the y scale) and in order to connect it dynamically to the other forward plots it is necessary to use function `mmdrsplot`.
init	Point where to start monitoring required diagnostics. If `init` is not specified it will be set equal to `(p+1)`.
bsbsteps	A vector which specifies for which steps of the forward search it is necessary to save the units forming subset for each random start. if `bsbsteps = 0` for each random start we store the units forming subset in all steps. The default is store the units forming subset in all steps if `n <= 500` else to store the units forming subset at step init and steps which are multiple of 100. For example, if `n = 753` and `init = 6`, units forming subset are stored for `m=init, 100, 200, 300, 400, 500` and `600`. REMARK: The vector bsbsteps must contain numbers from init to n. if `min(bsbsteps) < init` a warning message will be issued.
nsimul	Number of random starts. Default value is `nsimul=200`.
nocheck	It controls whether to perform checks on matrix Y. If `nocheck=TRUE`, no check is performed.
numpool	If `numpool > 1`, the routine automatically checks if the Parallel Computing Toolbox is installed and distributes the random starts over numpool parallel processes. If `numpool <= 1`, the random starts are run sequentially. By default, numpool is set equal to the number of physical cores available in the CPU (this choice may be inconvenient if other applications are running concurrently). The same happens if the numpool value chosen by the user exceeds the available number of cores. REMARK: up to R2013b, there was a limitation on the maximum number of cores that could be addressed by the parallel processing toolbox (8 and, more recently, 12). From R2014a, it is possible to run a local cluster of more than 12 workers.

REMARK: Unless you adjust the cluster profile, the default maximum number of workers is the same as the number of computational (physical) cores on the machine.

REMARK: In modern computers the number of logical cores is larger than the number of physical cores. By default, MATLAB is not using all logical cores because, normally, hyper-threading is enabled and some cores are reserved to this feature.

REMARK: It is because of Remarks 3 that we have chosen as default value for numpool the number of physical cores rather than the number of logical ones. The user can increase the number of parallel pool workers allocated to the multiple start monitoring by:

- setting the NumWorkers option in the local cluster profile settings to the number of logical cores (Remark 2). To do so go on the menu *Home|Parallel|Manage Cluster Profile* and set the desired "Number of workers to start on your local machine".
- setting numpool to the desired number of workers

Therefore, *if a parallel pool is not already open*, UserOption numpool (if set) overwrites the number of workers set in the local/current profile. Similarly, the number of workers in the local/current profile overwrites default value of `numpool` obtained as feature('numCores') (i.e. the number of physical cores).

`cleanpool` Set cleanpool `cleanpool=TRUE` if the parallel pool has to be cleaned after the execution of the random starts. Otherwise (default) `cleanpool=FALSE`. Clearly this option has an effect just if previous option `numpool > 1`.

`msg` Level of output to sidplay. It controls whether to display or not messages about random start progress. More precisely, if previous option `numpool > 1`, then a progress bar is displayed, on the other hand a message will be displayed on the screen when 10

REMARK: in order to create the progress bar when `nparpool > 1` the program writes on a temporary .txt file in the folder where the user is working. Therefore it is necessary to work in a folder where the user has write permission. If this is not the case and the user (say) is working without write permission in folder C:/Program Files/MATLAB the following message will appear on the screen:

Error using ProgressBar (line 57) Do you have write permissions for C:/Program Files/MATLAB?"

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

Returns an object of class `fsmmmdrs.object`.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C., Riani, M., and Cerioli, A. (2006), Random Start Forward Searches with Envelopes for Detecting Clusters in Multivariate Data, in: Zani S., Cerioli A., Riani M., Vichi M., Eds., *Data Analysis, Classification and the Forward Search*, pp. 163-172, Springer Verlag.

Atkinson, A.C. and Riani, M., (2007), Exploratory Tools for Clustering Multivariate Data, *Computational Statistics and Data Analysis*, Vol. 52, pp. 272-285, doi:10.1016/j.csda.2006.12.034

Riani, M., Cerioli, A., Atkinson, A.C., Perrotta, D. and Torti, F. (2008), Fitting Mixtures of Regression Lines with the Forward Search, in: *Mining Massive Data Sets for Security*, F. Fogelman-Soulie et al. Eds., pp. 271-286, IOS Press.

Examples

```
## Not run:
data(hbk, package="robustbase")
out <- fsmmmdrs(hbk[,1:3])
class(out)
summary(out)

## End(Not run)
```

`fsmmmdrs.object` *Description of* `fsmmmdrs.object` *Objects*

Description

An object of class `fsmmmdrs.object` holds information about the result of a call to `fsmmmdrs`.

Value

The object itself is basically a `list` with the following components:

`mmdrs`: Minimum Mahalanobis distance, (n-init) by (nsimul+1) matrix which contains the monitoring of minimum Mahalanobis distance at each step of the forward search.

- 1st column = fwd search index (from init to n-1)
- 2nd column = minimum Mahalanobis distance for random start 1
- 3rd column ...
- nsimul+1 column minimum Mahalanobis distance for random start nsimul

`BBrs`: Units belonging to the subset at the steps specified by input option bsbsteps. If `bsbsteps=0` `BBrs` has size `n-by-(n-init+1)-by-nsimul`. In this case `BBrs[,,j]` with j=1, 2, ..., nsimul has the following structure:

- 1st row = has number 1 in correspondence of the steps in which unit 1 is included inside subset and a missing value for the other steps
- ...

- (n-1)-th row = has number n-1 in correspondence of the steps in which unit n-1 is included inside subset and a missing value for the other steps
- n-th row = has the number n in correspondence of the steps in which unit n is included inside subset and a missing value for the other steps

If, on the other hand, bsbsteps is a vector which specifies the steps of the search in which it is necessary to store subset, BBrs has size `n-by-length(bsbsteps)-by-nsimul`. In other words, `BBrs[,,j]` with `j=1, 2, ..., nsimul` has the same structure as before, but now contains just `length(bsbsteps)` columns.

`X`: the data matrix X.

The object has class `"fsmmmdrs"`.

Examples

```
## Not run:
    data(hbk, package="robustbase")
    out <- fsmmmdrs(hbk[,1:3])
    class(out)
    summary(out)

## End(Not run)
```

`fsmult` *Gives an automatic outlier detection procedure in multivariate analysis*

Description

Gives an automatic outlier detection procedure in multivariate analysis and performs forward search in multivariate analysis with exploratory data

Usage

```
fsmult(
  x,
  bsb,
  monitoring = FALSE,
  crit = c("md", "biv", "uni"),
  rf = 0.95,
  init,
  plot = FALSE,
  bonflev,
  msg = TRUE,
  nocheck = FALSE,
  scaled = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
bsb	List of units forming the initial subset or size of the initial subset. If `monitoring=FALSE` the default is to start the search with `p+1` units, containing those observations which are not outlying on any scatterplot, found as the intersection of all points lying within a robust contour containing a specified portion of the data (Riani and Zani 1997) and inside the univariate boxplot. Remark: if bsb is a vector, the option crit is ignored.
monitoring	Wheather to perform monitoring of Mahalanobis distances and other specific quantities
crit	If specified, the criterion to be used to initialize the search. • If `crit="md"` the units which form initial subset are those which have the smallest `m0` pseudo Mahalanobis distances computed using procedure `unibiv()` (bivariate robust ellipses). • If `crit="biv"` sorting is done first in terms of times units fell outside robust bivariate ellipses and then in terms of pseudoMD. In other words, the units forming initial subset are chosen first among the set of those which never fell outside robust bivariate ellipses then among those which fell only once outside bivariate ellipses ... up to reach `m0`. • If `crit="uni"` sorting is done first in terms of times units fell outside univariate boxplots and then in terms of pseudoMD. In other words, the units forming initial subset are chosen first among the set of those which never fell outside univariate boxplots then among those which fell only once outside univariate boxplots... up to reach `m0`. Remark: as the user can see the starting point of the search is not going to affect at all the results of the analysis. The user can explore this point with his own datasets. Remark: if `crit="biv"` the user can also supply in scalar rf (see below) the confidence level of the bivariate ellipses.
rf	Confidence level for bivariate ellipses. The default is 0.95. This option is useful only if `crit='biv'`.
init	Point where to start monitoring required diagnostics. Note that if a vector `m0` is supplied, `init >= length(m0)`. If `init` is not specified it will be set equal to `floor(n*0.6)`.
plot	Plots the minimum Mahalanobis distance. If `plot=FALSE` (default) or `plot=0` no plot is produced. If `plot=TRUE` the plot of minimum MD with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plot=2` the additional plots of envelope resuperimposition are produced. If plot is a list it may contain the following fields: • ylim vector with two elements controlling minimum and maximum on the y axis. Default value is " (automatic scale)

- xlim vector with two elements controlling minimum and maximum on the x axis. Default value is '' (automatic scale)
- resuper vector which specifies for which steps it is necessary to show the plots of resuperimposed envelopes if resuper is not supplied a plot of each step in which the envelope is resuperimposed is shown. Example: if `resuper = c(85 87)` plots of resuperimposedenvelopes are shown at steps `m=85` and `m=87`
- ncoord If `ncoord=1` plots are shown in normal coordinates else (default) plots are shown in traditional mmd coordinates
- labeladd If `labeladd=1`, the outliers in the spm are labelled with the unit row index. The default value is `labeladd=""`, i.e. no label is added
- nameY character vector containing the labels of the variables. As default value, the labels which are added are Y1, ...Yp.
- lwd controls line width of the curve which contains the monitoring of minimum Mahalanobis distance. Default is `lwd=2`.
- lwdenv Controls linewidth of the envelopes. Default is `lwdenv=2`.

`bonflev` Option that might be used to identify extreme outliers when the distribution of the data is strongly non normal. In these circumstances, the general signal detection rule based on consecutive exceedances cannot be used. In this case `bonflev` can be:

1. a scalar smaller than 1, which specifies the confidence level for a signal and a stopping rule based on the comparison of the minimum deletion residual with a Bonferroni bound. For example if `bonflev=0.99` the procedure stops when the trajectory exceeds for the first time the 99 per cent bonferroni bound.
2. a scalar value greater than 1. In this case the procedure stops when the residual trajectory exceeds for the first time this value.

Default value is empty, which means to rely on general rules based on consecutive exceedances.

`msg` It controls whether to display or not messages on the screen. If `msg=TRUE` (default) messages about the progression of the search are displayed on the screen otherwise only error messages will be displayed.

`nocheck` It controls whether to perform checks on matrix Y. If `nocheck=TRUE`, no check is performed.

`scaled` Controls whether to monitor scaled Mahalanobis distances (only if `monitoring=TRUE`). If `scaled=TRUE` Mahalanobis distances monitored during the search are scaled using ratio of determinant. If `scaled=2` Mahalanobis distances monitored during the search are scaled using asymptotic consistency factor. The default is `scaled=FALSE`, that is Mahalanobis distances are not scaled.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

Depending on the input parameter `monitoring`, one of the following objects will be returned:

1. fsmult.object
2. fsmeda.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Riani, M., Atkinson A.C., Cerioli A. (2009). Finding an unknown number of multivariate outliers. Journal of the Royal Statistical Society Series B, Vol. 71, pp. 201-221.

Cerioli A., Farcomeni A., Riani M., (2014). Strong consistency and robustness of the Forward Search estimator of multivariate location and scatter, Journal of Multivariate Analysis, Vol. 126, pp. 167-183, http://dx.doi.org/10.1016/j.jmva.2013.12.010.

Atkinson Riani and Cerioli (2004), *Exploring multivariate data with the forward search* Springer Verlag, New York.

Examples

```
## Not run:

data(hbk, package="robustbase")
(out <- fsmult(hbk[,1:3]))
class(out)
summary(out)

##  Generate contaminated data (200,3)
n <- 200
p <- 3
set.seed(123456)
X <- matrix(rnorm(n*p), nrow=n)
Xcont <- X
Xcont[1:5, ] <- Xcont[1:5,] + 3

out1 <- fsmult(Xcont, trace=TRUE)           # no plots (plot defaults to FALSE)
names(out1)

(out1 <- fsmult(Xcont, trace=TRUE, plot=TRUE))    # identical to plot=1

## plot=1 - minimum MD with envelopes based on n observations
##  and the scatterplot matrix with the outliers highlighted
(out1 <- fsmult(Xcont, trace=TRUE, plot=1))

## plot=2 - additional plots of envelope resuperimposition
(out1 <- fsmult(Xcont, trace=TRUE, plot=2))

## plots is a list: plots showing envelope superimposition in normal coordinates.
(out1 <- fsmult(Xcont, trace=TRUE, plot=list(ncoord=1)))

##  Choosing an initial subset formed by the three observations with
##  the smallest Mahalanobis Distance.
```

```
(out1 <- fsmult(Xcont, m0=5, crit="md", trace=TRUE))

## fsmult() with monitoring
(out2 <- fsmult(Xcont, monitoring=TRUE, trace=TRUE))
names(out2)

## Monitor the exceedances from m=200 without showing plots.
n <- 1000
p <- 10
Y <- matrix(rnorm(10000), ncol=10)
(out <- fsmult(Y, init=200))

##  Forgery Swiss banknotes examples.

data(swissbanknotes)

## Monitor the exceedances of Minimum Mahalanobis Distance
(out1 <- fsmult(swissbanknotes[101:200,], plot=1))

##  Control minimum and maximum on the x axis
(out1 <- fsmult(swissbanknotes[101:200,], plot=list(xlim=c(60,90))))

##  Monitor the exceedances of Minimum Mahalanobis Distance using
##  normal coordinates for mmd.
(out1 <- fsmult(swissbanknotes[101:200,], plot=list(ncoord=1)))

## End(Not run)
```

`fsmult.object` *Description of* `fsmult.object` *Objects*

Description

An object of class `fsmult.object` holds information about the result of a call to `fsmult`.

Value

The object itself is basically a `list` with the following components:

`outliers`	kx1 vector containing the list of the k units declared as outliers or NULL if the sample is homogeneous.
`loc`	p-by-1 vector containing location of the data.
`cov`	p-by-p robust estimate of covariance matrix.
`md`	n-by-1 vector containing the estimates of the robust Mahalanobis distances (in squared units). This vector contains the distances of each observation from the location of the data, relative to the scatter matrix cov.
`mmd`	(n-init)-by-2 matrix. 1st col is the forward search index; 2nd col is the value of minimum Mahalanobis Distance in each step of the fwd search.

Un	(n-init)-by-11 matrix which contains the unit(s) included in the subset at each step of the fwd search. REMARK: in every step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one. Un[1 ,2] for example contains the unit included in step init+1. Un[end, 2] contains the units included in the final step of the search.
nout	2 x 5 matrix containing the number of times mdr went out of particular quantiles. First row contains quantiles 1 99 99.9 99.99 99.999. Second row contains the frequency distribution. It is NULL if bonflev threshold is used.
constr	This output is produced only if the search found at a certain step is a non singular matrix X. In this case the search run in a constrained mode, that is including the units which produced a singular matrix in the last n-constr steps. out.constr is a vector which contains the list of units which produced a singular X matrix.
X	the data matrix X

The object has class "fsmult".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsmult(hbk[,1:3]))
    class(out)
    summary(out)

## End(Not run)
```

---

fsr.object *Description of* fsr *Objects*

Description

An object of class fsr.object holds information about the result of a call to fsreg.

Value

The object itself is basically a list with the following components:

beta	p-by-1 vector containing the estimated regression parameters (in step n-k).
scale	scalar containing the estimate of the scale (sigma).
residuals	residuals.
fittedvalues	fitted values.
outliers	kx1 vector containing the list of the k units declared as outliers or NULL if the sample is homogeneous.
mdr	(n-init) x 2 matrix 1st col = fwd search index, 2nd col = value of minimum deletion residual in each step of the fwd search

Un	(n-init) x 11 matrix which contains the unit(s) included in the subset at each step of the fwd search. REMARK: in every step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one. Un(1,2) for example contains the unit included in step init+1. Un(end,2) contains the units included in the final step of the search.
nout	2 x 5 matrix containing the number of times mdr went out of particular quantiles. First row contains quantiles 1 99 99.9 99.99 99.999. Second row contains the frequency distribution.
constr	This output is produced only if the search found at a certain step is a non singular matrix X. In this case the search run in a constrained mode, that is including the units which produced a singular matrix in the last n-constr steps. out.constr is a vector which contains the list of units which produced a singular X matrix.
X	the data matrix X
y	the response vector y

The object has class "fsr".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="FS"))
    class(out)
    summary(out)

## End(Not run)
```

FSR_control *Creates an* FSR_control *object*

Description

Creates an object of class FSR_control to be used with the fsreg() function, containing various control parameters.

Usage

```
FSR_control(intercept = TRUE, h, nsamp = 1000, lms = 1, init, nocheck = FALSE,
    bonflev = "", msg = TRUE, bsbmfullrank = TRUE,
    plot = FALSE, bivarfit = FALSE, multivarfit = FALSE,
    labeladd = FALSE, nameX, namey, ylim, xlim)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
h	The number of observations that have determined the least trimmed squares estimator, scalar. h is an integer greater or equal than p but smaller then n. Generally if the purpose is outlier detection h=[0.5*(n+p+1)] (default value). h can be smaller than this threshold if the purpose is to find subgroups of homogeneous observations. In this function the LTS/LMS estimator is used just to initialize the search.
nsamp	Number of subsamples which will be extracted to find the robust estimator, scalar. If nsamp=0 all subsets will be extracted. They will be (n choose p). If the number of all possible subset is <1000 the default is to extract all subsets otherwise just 1000.
lms	Criterion to use to find the initial subset to initialize the search (LMS, LTS with concentration steps, LTS without concentration steps or subset supplied directly by the user). The default value is 1 (Least Median of Squares is computed to initialize the search). On the other hand, if the user wants to initialze the search with LTS with all the default options for concentration steps then lms=2. If the user wants to use LTS without concentration steps, lms can be a scalar different from 1 or 2. If lms is a list it is possible to control a series of options for concentration steps (for more details see option lms inside LXS_control). If, on the other hand, the user wants to initialize the search with a prespecified set of units there are two possibilities: 1. lms can be a vector with length greater than 1 which contains the list of units forming the initial subset. For example, if the user wants to initialize the search with units 4, 6 and 10 then lms=c(4, 6, 10); 2. lms is a struct which contains a field named bsb which contains the list of units to initialize the search. For example, in the case of simple regression through the origin with just one explanatory variable, if the user wants to initialize the search with unit 3 then lms=list(bsb=3).
init	Search initialization, scalar. It specifies the initial subset size to start monitoring exceedances of minimum deletion residual, if init is not specified it set equal to: p+1, if the sample size is smaller than 40 or min(3*p+1,floor(0.5*(n+p+1))), otherwise. For example, if init=100, the procedure starts monitoring from step m=100.
nocheck	Check input arguments, scalar. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. As default nocheck=FALSE.
bonflev	Option to be used if the distribution of the data is strongly non normal and, thus, the general signal detection rule based on consecutive exceedances cannot be used. In this case bonflev can be: 1. a scalar smaller than 1 which specifies the confidence level for a signal and a stopping rule based on the comparison of the minimum MD with a Bonferroni bound. For example if bonflev=0.99 the procedure stops when the trajectory exceeds for the first time the 99% bonferroni bound.

2. A scalar value greater than 1. In this case the procedure stops when the residual trajectory exceeds for the first time this value.

Default value is '', which means to rely on general rules based on consecutive exceedances.

`msg` Controls whether to display or not messages on the screen If `msg==1` (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

`bsbmfullrank` How to behave in case subset at step m (say bsbm) produces a singular X. In other words, this options controls what to do when `rank(X[bsbm, ])` is smaller then number of explanatory variables. If `bsbmfullrank=1` (default) these units (whose number is say mnofullrank) are constrained to enter the search in the final n-mnofullrank steps else the search continues using as estimate of beta at step m the estimate of beta found in the previous step.

`plot` Plot on the screen. Scalar. If `plot=TRUE` the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plot=2` the user can also monitor the intermediate plots based on envelope superimposition. If `plot=FALSE` (default) no plot is produced.

`bivarfit` Wheather to superimpose bivariate least square lines on the plot (if `plot=TRUE`. This option adds one or more least squares lines, based on SIMPLE REGRESSION of y on Xi, to the plots of y|Xi. The default is `bivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. If `bivarfit=2`, two OLS lines are fitted: one to all points and another to the group of the genuine observations. The group of the potential outliers is not fitted. If `bivarfit=0` one OLS line is fitted to each group. This is useful for the purpose of fitting mixtures of regression lines. If `bivarfit='i1'` or `bivarfit='i2'`, etc. an OLS line is fitted to a specific group, the one with index 'i' equal to 1, 2, 3 etc. Again, useful in case of mixtures.

`multivarfit` Wheather to superimpose multivariate least square lines. This option adds one or more least square lines, based on MULTIVARIATE REGRESSION of y on X, to the plots of y|Xi. The default is `multivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. The line added to the scatter plot y|Xi is avconst + Ci*Xi, where Ci is the coefficient of Xi in the multivariate regression and avconst is the effect of all the other explanatory variables different from Xi evaluated at their centroid (that is overline(y)'C)). If `multivarfit=2`, same action as with `multivarfit=1` but this time we also add the line based on the group of unselected observations (i.e. the normal units).

`labeladd` Add outlier labels in plot. If `labeladd=TRUE`, we label the outliers with the unit row index in matrices X and y. The default value is `labeladd=FALSE`, i.e. no label is added.

`nameX` Add variable labels in plot. A vector of strings of length p containing the labels of the variables of the regression dataset. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

`namey` Add response label. A string containing the label of the response

ylim	Control y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
xlim	Control x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.

Details

Creates an object of class FSR_control to be used with the fsreg() function, containing various control parameters.

Value

An object of class "FSR_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also Sreg_control, MMreg_control, LXS_control, FSReda_control, Sregeda_control and MMregeda_control.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="FS", control=FSR_control(h=56, nsamp=500, lms=2)))
summary(out)

## End(Not run)
```

---

fsrbase *fsrbase: an automatic outlier detection procedure in linear regression*

Description

An automatic outlier detection procedure in linear regression

Usage

```
fsrbase(x, ...)

## S3 method for class 'formula'
fsrbase(formula, data, subset, weights, na.action,
       model = TRUE, x.ret = FALSE, y.ret = FALSE,
       contrasts = NULL, offset, ...)
```

```
## Default S3 method:
fsrbase(x, y, bsb, intercept = TRUE,
        monitoring = FALSE, control, trace = FALSE,
        ...)
```

Arguments

formula	a formula of the form y ~ x1 + x2 + ...
data	data frame from which variables specified in formula are to be taken.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. **NOT USED YET**.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The "factory-fresh" default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
model, x.ret, y.ret	logicals indicating if the model frame, the model matrix and the response are to be returned, respectively.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. An offset term can be included in the formula instead or as well, and if both are specified their sum is used.
x	Predictor variables. Matrix. Matrix of explanatory variables (also called 'regressors') of dimension n x (p-1) where p denotes the number of explanatory variables including the intercept. Rows of X represent observations, and columns represent variables. By default, there is a constant term in the model, unless you explicitly remove it using input option intercept=FALSE, so do not include a column of 1s in X. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
y	Response variable. Vector. Response variable, specified as a vector of length n, where n is the number of observations. Each entry in y is the response for the corresponding row of X. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
bsb	Initial subset - vector of indices. If bsb=0 (default) then the procedure starts with p units randomly chosen. If bsb is not 0 the search will start with m0=length(bsb).
intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
monitoring	wheather to perform monitoring for several quantities in each step of the forward search. Deafault is monitoring=FALSE.

control	A control object (S3) containing estimation options, as returned by FSR_control. Use the function FSR_control and see its help page. If the control object is supplied, the parameters from it will be used. If parameters are passed also in the invocation statement, they will override the corresponding elements of the control object.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	Potential further optional arguments, see the help of the function FSR_control.

Value

Depending on the input parameter monitoring, one of the following objects will be returned:

1. fsr.object
2. fsreda.object

Author(s)

FSDA team

References

Riani, M., Atkinson A.C., Cerioli A. (2009). Finding an unknown number of multivariate outliers. Journal of the Royal Statistical Society Series B, Vol. 71, pp. 201-221.

Examples

```
## Not run:

n <- 200
p <- 3

X <- matrix(data=rnorm(n*p), nrow=n, ncol=p)
y <- matrix(data=rnorm(n*1), nrow=n, ncol=1)
(out = fsrbase(X, y))

## Now we use the formula interface:
(out1 = fsrbase(y~X, control=FSR_control(plot=FALSE)))

## Or use the variables in a data frame
(out2 = fsrbase(Y~., data=hbk, control=FSR_control(plot=FALSE)))

## let us compare to the LTS solution
(out3 = ltsReg(Y~., data=hbk))

## Now compute the model without intercept
(out4 = fsrbase(Y~.-1, data=hbk, control=FSR_control(plot=FALSE)))

## And compare again with the LTS solution
(out5 = ltsReg(Y~.-1, data=hbk))

## using default (optional arguments)
```

```
    (out6 = fsrbase(Y~.-1, data=hbk, control=FSR_control(plot=FALSE, nsamp=1500, h=50)))

## End(Not run)
```

`fsreda.object` *Description of* `fsreda` *Objects*

Description

An object of class `fsreda.object` holds information about the result of a call to `fsreg`.

Value

The object itself is basically a `list` with the following components:

`RES` n x (n-init+1) matrix containing the monitoring of scaled residuals: the first row is the residual for the first unit, ..., n-th row is the residual for the n-th unit.

`LEV` (n+1) x (n-init+1) matrix containing the monitoring of leverage: the first row is the leverage for the first unit, ..., n-th row is the leverage for the n-th unit.

`BB` n x (n-init+1) matrix containing the information about the units belonging to the subset at each step of the forward search: first col contains indexes of the units forming subset in the initial step; ...; last column contains units forming subset in the final step (all units).

`mdr` n-init x 3 matrix which contains the monitoring of minimum deletion residual or (m+1)-ordered residual at each step of the forward search: first col is the fwd search index (from init to n-1); 2nd col = minimum deletion residual; 3rd col = (m+1)-ordered residual.

Remark: these quantities are stored with sign, that is the min deletion residual is stored with negative sign if it corresponds to a negative residual.

`msr` n-init+1 x 3 matrix which contains the monitoring of maximum studentized residual or m-th ordered residual: first col is the fwd search index (from init to n); 2nd col = maximum studentized residual; 3rd col = (m)-ordered studentized residual.

`nor` (n-init+1) x 4 matrix containing the monitoring of normality test in each step of the forward search: first col = fwd search index (from init to n); 2nd col = Asymmetry test; 3rd col = Kurtosis test; 4th col = Normality test.

`Bols` (n-init+1) x (p+1) matrix containing the monitoring of estimated beta coefficients in each step of the forward search.

`S2` (n-init+1) x 5 matrix containing the monitoring of S2 or R2 and F-test in each step of the forward search:

1. 1st col = fwd search index (from init to n);
2. 2nd col = monitoring of S2;
3. 3rd col = monitoring of R2;

4. 4th col = monitoring of rescaled S2. In this case the estimated of sigma^2 at step m is divided by the consistency factor (to make the estimate asymptotically unbiased)
5. 5th col = monitoring of F test. Note that an asymptotic unbiased estimate of sigma^2 is used.

In this case the estimated of s2 at step m is divided by the consistency factor (to make the estimate asymptotically unbiased).

`coo` (n-init+1) x 3 matrix containing the monitoring of Cook or modified Cook distance in each step of the forward search:

1. 1st col = fwd search index (from init to n);
2. 2nd col = monitoring of Cook distance;
3. 3rd col = monitoring of modified Cook distance.

`Tols` (n-init+1) x (p+1) matrix containing the monitoring of estimated t-statistics (as specified in option input 'tstat') in each step of the forward search

`Un` (n-init) x 11 matrix which contains the unit(s) included in the subset at each step of the fwd search.

REMARK: in every step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one Un(1,2) for example contains the unit included in step init+1 Un(end,2) contains the units included in the final step of the search.

`betaINT` Confidence intervals for the elements of β. betaINT is a (n-init+1)-by-2*length(confint)-by-p 3D array. Each third dimension refers to an element of beta:

1. `betaINT[,,1]` is associated with first element of beta;
2. ...;
3. `betaINT[,,p]` is associated with last element of beta.

The first two columns contain the lower and upper confidence limits associated with conflev(1). Columns three and four contain the lower and upper confidence limits associated with conflev(2); ...; The last two columns contain the lower and upper confidence limits associated with conflev(end). For example `betaINT[,3:4,5]` contain the lower and upper confidence limits for the fifth element of beta using confidence level specified in the second element of input option `conflev`.

`sigma2INT` confidence interval for s2.

1. 1st col = fwd search index;
2. 2nd col = lower confidence limit based on conflev(1);
3. 3rd col = upper confidence limit based on conflev(1);
4. 4th col = lower confidence limit based on conflev(2);
5. 5th col = upper confidence limit based on conflev(2);
6. ...
7. penultimate col = lower confidence limit based on conflev(end);
8. last col = upper confidence limit based on conflev(end).

`X` the data matrix X

`y` the response vector y

The object has class `"fsreda"`.

Examples

```
 ## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="FS", monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

FSReda_control *Creates an* FSReda_control *object*

Description

Creates an object of class FSReda_control to be used with the fsreg() function, containing various control parameters.

Usage

```
FSReda_control(intercept = TRUE, init, nocheck = FALSE,
    tstat = c("trad", "scal"), conflev = c(0.95, 0.99))
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
init	Search initialization, scalar. It specifies the initial subset size to start monitoring exceedances of minimum deletion residual, if init is not specified it set equal to: p+1, if the sample size is smaller than 40 or min(3*p+1,floor(0.5*(n+p+1))), otherwise. For example, if init=100, the procedure starts monitoring from step m=100.
nocheck	Check input arguments, scalar. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. As default nocheck=FALSE.
tstat	The kind of t-statistics which have to be monitored. tstat="trad" implies monitoring of traditional t statistics (out$Tols). In this case the estimate of s2 at step m is based on s2m (notice that s2m«s2 when m/n is small) tstat="scal" (default) implies monitoring of rescaled t statistics. In this case the estimate of s2 at step m is based on s2m/vartruncnorm(m/n) where vartruncnorm(m/n) is the variance of the truncated normal distribution.
conflev	Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975.

Details

Creates an object of class FSReda_control to be used with the fsreg() function, containing various control parameters.

Value

An object of class "FSReda_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as FSR_control, MMreg_control and LXS_control

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="FS", monitoring=TRUE,
    control=FSReda_control(tstat="scal")))

## End(Not run)
```

---

fsreg *fsreg: an automatic outlier detection procedure in linear regression*

Description

An automatic outlier detection procedure in linear regression

Usage

```
fsreg(x, ...)

## S3 method for class 'formula'
fsreg(formula, data, subset, weights, na.action,
       model = TRUE, x.ret = FALSE, y.ret = FALSE,
       contrasts = NULL, offset, ...)

## Default S3 method:
fsreg(x, y, bsb, intercept = TRUE,
        family = c("homo", "hetero", "bayes"),
  method = c("FS", "S", "MM", "LTS", "LMS"),
       monitoring = FALSE, control, trace = FALSE,
       ...)
```

Arguments

formula	a formula of the form y ~ x1 + x2 + ...
data	data frame from which variables specified in formula are to be taken.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. **NOT USED YET**.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The "factory-fresh" default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
model, x.ret, y.ret	logicals indicating if the model frame, the model matrix and the response are to be returned, respectively.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. An offset term can be included in the formula instead or as well, and if both are specified their sum is used.
family	family of robust regression models, can be 'homo' for homoscedastic (same variance) regression model, 'hetero' for heteroskedastic regression model or 'bayes' Bayesian linear regression. The default is family='homo' for homoscedastic regression model.
method	robust regression estimation model, can be 'FS' for Forward search, 'S' for S regression, 'MM' for MM regression, 'LMS' or 'LTS'. The default is method='FS' for forward search estimation.
monitoring	wheather to perform monitoring for several quantities in each step of the forward search or for series of values of the breakdown point in case of S estimates or for series of values of the efficiency in case of MM estimates. Deafault is monitoring=FALSE.
y	Response variable. Vector. Response variable, specified as a vector of length n, where n is the number of observations. Each entry in y is the response for the corresponding row of X. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
x	Predictor variables. Matrix. Matrix of explanatory variables (also called 'regressors') of dimension n x (p-1) where p denotes the number of explanatory variables including the intercept. Rows of X represent observations, and columns represent variables. By default, there is a constant term in the model, unless you explicitly remove it using input option intercept=FALSE, so do not include a column of 1s in X. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
bsb	Initial subset - vector of indices. If bsb=0 (default) then the procedure starts with p units randomly chosen. If bsb is not 0 the search will start with m0=length(bsb).

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
control	A control object (S3) containing estimation options. If the control object is supplied, the parameters from it will be used. If parameters are passed also in the invocation statement, they will override the corresponding elements of the control object.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

Depending on the input parameters family and method, one of the following objects will be returned:

1. fsr.object
2. sreg.object
3. mmreg.object
4. fsdalms.object
5. fsdalts.object
6. fsreda.object
7. sregeda.object
8. mmregeda.object

Author(s)

FSDA team

References

Riani, M., Atkinson A.C., Cerioli A. (2009). Finding an unknown number of multivariate outliers. Journal of the Royal Statistical Society Series B, Vol. 71, pp. 201-221.

Examples

```
## Not run:

library(robustbase)

n <- 200
p <- 3

X <- matrix(data=rnorm(n*p), nrow=n, ncol=p)
y <- matrix(data=rnorm(n*1), nrow=n, ncol=1)
(out = fsreg(X, y))

## Now we use the formula interface:
(out1 = fsreg(y~X, control=FSR_control(plot=FALSE)))
```

```
## Or use the variables in a data frame
(out2 = fsreg(Y~., data=hbk, control=FSR_control(plot=FALSE)))

## let us compare to the LTS solution
library(robustbase)
(out3 = ltsReg(Y~., data=hbk))

## Now compute the model without intercept
(out4 = fsreg(Y~.-1, data=hbk, control=FSR_control(plot=FALSE)))

## And compare again with the LTS solution
(out5 = ltsReg(Y~.-1, data=hbk))

## using default (optional arguments)
(out6 = fsreg(Y~.-1, data=hbk, control=FSR_control(plot=FALSE, nsamp=1500, h=50)))

## End(Not run)
```

`fsrfan` *Robust transformations for regression*

Description

The transformations for negative and positive responses were determined by Yeo and Johnson (2000) by imposing the smoothness condition that the second derivative of zYJ (λ) with respect to y be smooth at y = 0. However some authors, for example Weisberg (2005), query the physical interpretability of this constraint which is oftern violated in data analysis. Accordingly, Atkinson et al. (2019) and (2020) extend the Yeo-Johnson transformation to allow two values of the transformations parameter: λ_N for negative observations and λ_P for non-negative ones.

FSRfan monitors:

1. the t test associated with the constructed variable computed assuming the same transformation parameter for positive and negative observations fixed. In short we call this test, "global score test for positive observations".
2. the t test associated with the constructed variable computed assuming a different transformation for positive observations keeping the value of the transformation parameter for negative observations fixed. In short we call this test, "test for positive observations".
3. the t test associated with the constructed variable computed assuming a different transformation for negative observations keeping the value of the transformation parameter for positive observations fixed. In short we call this test, "test for negative observations".
4. the F test for the joint presence of the two constructed variables described in points 2) and 3).
5. the F likelihood ratio test based on the MLE of λ_P and λ_N. In this case the residual sum of squares of the null model bsaed on a single trasnformation parameter λ is compared with the residual sum of squares of the model based on data transformed data using MLE of λ_P and λ_N.

Usage

```
fsrfan(x, ...)

## S3 method for class 'formula'
fsrfan(
  formula,
  data,
  subset,
  weights,
  na.action,
  model = TRUE,
  x.ret = FALSE,
  y.ret = FALSE,
  contrasts = NULL,
  offset,
  ...
)

## Default S3 method:
fsrfan(
  x,
  y,
  intercept = TRUE,
  plot = FALSE,
  family = c("BoxCox", "YJ", "YJpn", "YJall"),
  la = c(-1, -0.5, 0, 0.5, 1),
  lms,
  alpha = 0.75,
  h,
  init,
  msg = FALSE,
  nocheck = FALSE,
  nsamp = 1000,
  conflev = 0.99,
  xlab,
  ylab,
  main,
  xlim,
  ylim,
  lwd = 2,
  lwd.env = 1,
  trace = FALSE,
  ...
)

## S3 method for class 'fsrfan'
plot(
  x,
```

```
  conflev = 0.99,
  xlim,
  ylim,
  xlab = "Subset of size m",
  ylab = "Score test statistic",
  main = "Fan plot",
  col,
  lty,
  lwd = 2.5,
  lwd.env = 1,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
...	potential further arguments passed to lower level functions.
formula	a formula of the form y ~ x1 + x2 + ...
data	data frame from which variables specified in formula are to be taken.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used **NOT USED YET**.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The "factory-fresh" default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
model	logical indicating if the model frame, is to be returned.
x.ret	logical indicating if the the model matrixis to be returned.
y.ret	logical indicating if the response is to be returned.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. An offset term can be included in the
y	Response variable. A vector with n elements that contains the response variable.
intercept	wheather to use constant term (default is intercept=TRUE
plot	If plot=FALSE (default) no plot is produced. If plot=TRUE a fan plot is shown.
family	string which identifies the family of transformations which must be used. Possible values are c('BoxCox', 'YJ', 'YJpn', 'YJall'). Default is 'BoxCox'. The Box-Cox family of power transformations equals $(y^\lambda - 1)/\lambda$ for λ not equal to zero, and $\log(y)$ if $\lambda = 0$. The Yeo-Johnson (YJ) transformation is the Box-Cox transformation of $y + 1$ for nonnegative values, and of $\|y\| + 1$

with parameter $2 - \lambda$ for y negative. Remember that BoxCox can be used only if input y is positive. Yeo-Johnson family of transformations does not have this limitation. If `family='YJpn'` Yeo-Johnson family is applied but in this case it is also possible to monitor (in the output arguments `Scorep` and `Scoren`) the score test for positive and negative observations respectively. If `family='YJall'`, it is also possible to monitor the joint F test for the presence of the two constructed variables for positive and negative observations.

`la` values of the transformation parameter for which it is necessary to compute the score test. Default value of lambda is `la=c(-1, -0.5, 0, 0.5, 1)`, i.e., the five most common values of lambda.

`lms` how to find the initlal subset to initialize the search. If `lms=1` (default) Least Median of Squares (LMS) is computed, else Least Trimmed Squares (LTS) is computed. If, `lms` is matrix of size `p - 1 + intercept` `X length(la)` it contains in column `j=1, . . . , lenght(la)` the list of units forming the initial subset for the search associated with `la(j)`. In this case the input option `nsamp` is ignored.

`alpha` the percentage (roughly) of squared residuals whose sum will be minimized, by default `alpha=0.5`. In general, alpha must between 0.5 and 1.

`h` The number of observations that have determined the least trimmed squares estimator, scalar. `h` is an integer greater or equal than `p` but smaller then `n`. Generally `h=[0.5*(n+p+1)]` (default value).

`init` Search initialization. It specifies the initial subset size to start monitoring the value of the score test. If `init` is not specified it will be set equal to: `p+1`, if the sample size is smaller than 40 or `min(3 * p + 1, floor(0.5 * (n+p+1)))`, otherwise.

`msg` Controls whether to display or not messages on the screen. If `msg==TRUE` messages are displayed on the screen. If `msg=2`, detailed messages are displayed, for example the information at iteration level.

`nocheck` Whether to check input arguments. If `nocheck=TRUE` no check is performed on matrix `y` and matrix `X`. Notice that `y` and `X` are left unchanged. In other words the additional column of ones for the intercept is not added. The default is `nocheck=FALSE`.

`nsamp` number of subsamples which will be extracted to find the robust estimator. If `nsamp=0` all subsets will be extracted. They will be `n choose p`.

Remark: if the number of all possible subset is <1000 the default is to extract all subsets otherwise just 1000. If `nsamp` is a matrix of size `r-by-p`, it contains in the rows the subsets which sill have to be extracted. For example, if `p=3` and `nsamp=c(2,4,9; 23, 45, 49; 90, 34, 1)` the first subset is made up of units `c(2, 4, 9)`, the second subset of units `c(23, 45, 49)` and the third subset of units `c(90 34 1)`.

`conflev` Confidence level for the bands (default is 0.99, that is, we plot two horizontal lines corresponding to values -2.58 and 2.58).

`xlab` A label for the X-axis, default is 'Subset size m'

`ylab` A label for the Y-axis, default is 'Score test statistic'

`main` A label for the title, default is 'Fan plot'

`xlim` Minimum and maximum for the X-axis

ylim	Minimum and maximum for the Y-axis
lwd	The line width of the curves which contain the score test, a positive number, default is lwd=2
lwd.env	The line width of the lines associated with the envelopes, a positive number, default is lwd.env=1
trace	Whether to print intermediate results. Default is trace=FALSE.
col	a vector specifying the colors for the lines, each one corresponding to a la value. if length(col) < length(la), the colors will be recycled.
lty	a vector specifying the line types for the lines, each one corresponding to a la value. if length(col) < length(la), the colors will be recycled.

Value

An S3 object of class fsrfan.object will be returned which is basically a list containing the following elements:

1. la: vector containing the values of lambda for which fan plot is constructed
2. bs: matrix of size p X length(la) containing the units forming the initial subset for each value of lambda
3. Score: a matrix containing the values of the score test for each value of the transformation parameter:
   - 1st col = fwd search index;
   - 2nd col = value of the score test in each step of the fwd search for la[1]
   - ...
4. Scorep: matrix containing the values of the score test for positive observations for each value of the transformation parameter.

   Note: this output is present only if input option family='YJpn' or family='YJall'.
5. Scoren: matrix containing the values of the score test for negative observations for each value of the transformation parameter.

   Note: this output is present only if input option 'family' is 'YJpn' or 'YJall'.
6. Scoreb: matrix containing the values of the score test for the joint presence of both constructed variables (associated with positive and negative observations) for each value of the transformation parameter. In this case the reference distribution is the F with 2 and subset_size - p degrees of freedom.

   Note: this output is present only if input option family='YJall'.
7. Un: a three-dimensional array containing length(la) matrices of size retnUn=(n-init) X retpUn=11. Each matrix contains the unit(s) included in the subset at each step in the search associated with the corresponding element of la.

   REMARK: at each step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C. and Riani, M. (2000), *Robust Diagnostic Regression Analysis* Springer Verlag, New York.

Atkinson, A.C. and Riani, M. (2002), Tests in the fan plot for robust, diagnostic transformations in regression, *Chemometrics and Intelligent Laboratory Systems*, **60**, pp. 87–100.

Atkinson, A.C. Riani, M. and Corbellini A. (2019), The analysis of transformations for profit-and-loss data, *Journal of the Royal Statistical Society, Series C, "Applied Statistics"*, **69**, pp. 251–275. doi:10.1111/rssc.12389

Atkinson, A.C. Riani, M. and Corbellini A. (2021), The Box-Cox Transformation: Review and Extensions, *Statistical Science*, **36**(2), pp. 239–255. doi:10.1214/20STS778.

Examples

```
## Not run:
   data(wool)
   XX <- wool
   y <- XX[, ncol(XX)]
   X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

   out <- fsrfan(X, y)                   # call 'fsrfan' with all default parameters
   out <- fsrfan(cycles~., data=wool)    # use the formula interface

   set.seed(10)
   out <- fsrfan(cycles~., data=wool, plot=TRUE) # call 'fsrfan' and produce the plot
   plot(out)                               # use the plot method on the fsrfan object
  plot(out, conflev=c(0.9, 0.95, 0.99))   # change the confidence leel in the plot method

 ##======================
 ##
 ##  fsrfan() with all default options.

 ##  Store values of the score test statistic for the five most common
 ##  values of $\lambda$. Produce also a fan plot and display it on the screen.
 ##  Common part to all examples: load 'wool' data set.
 data(wool)
 head(wool)
 dim(wool)
 ##  The function fsrfan() stores the score test statistic.
 ##  In this case we use the five most common values of lambda are considered

 out <- fsrfan(cycles~., data=wool)
 plot(out)
 ##  fanplot(out)               # Not yet implemented in fsdaR

 ##  The fan plot shows the log transformation is diffused throughout the data
 ##  and does not depend on the presence of particular observations.
 ##======================
 ##
 ##  Example specifying 'lambda'.
 ##      Produce a fan plot for each value of 'lambda' in the vector 'la'.
```

```
##      Extract in matrix 'Un' the units which entered the search in each step

    data(wool)
    out <- fsrfan(cycles~., data=wool, la=c(-1, -0.5, 0, 0.5), plot=TRUE)
    plot(out)

    out$Un[,2,]

##======================
##  Example specifying the confidence level and the initial starting point for monitoring.
##  Construct the fan plot specifying the confidence level and the initial starting point
##   for monitoring.
    data(wool)
   out <- fsrfan(cycles~., data=wool, init=ncol(wool)+1, nsamp=0, conflev=0.95, plots=TRUE)
    plot(out, conflev=0.95)

##======================
##  Example with starting point based on LTS.
##  Extract all subsamples, construct a fan plot specifying the confidence level
##  and the initial starting point for monitoring based on p+2 observations,
##  strong line width for lines associated with the confidence bands.
    data(wool)
    out <- fsrfan(cycles~., data=wool, init=ncol(wool)+1, nsamp=0, lms=0,
        lwd.env=3, plot=TRUE)
    plot(out, lwd.env=3)

##======================
##  Fan plot using the loyalty cards data.
##  In this example, 'la' is the vector contanining the most common values
##  of the transformation parameter.
##  Store the score test statistics for the specified values of lambda
##  and automatically produce the fan plot
    data(loyalty)
    head(loyalty)
    dim(loyalty)

##  la is a vector contanining the most common values of the transformation parameter
    out <- fsrfan(amount_spent~., data=loyalty, la=c(0, 1/3, 0.4, 0.5),
          init=ncol(loyalty)+1, plot=TRUE, lwd=3)
    plot(out, lwd=3)

##  The fan plot shows that even if the third root is the best value of the transformation
##  parameter at the end of the search, in earlier steps it lies very close to the upper
##  rejection region. The best value of the transformation parameter seems to be the one
##  associated with la=0.4, which is always the confidence bands but at the end of search,
##  due to the presence of particular observations it goes below the lower rejection line.

##======================
##  Compare BoxCox with Yeo and Johnson transformation.
##  Store values of the score test statistic for the five most common
##  values of lambda. Produce also a fan plot and display it on the screen.
##  Common part to all examples: load wool dataset.
```

```
data(wool)

##  Store the score test statistic using Box Cox transformation.
outBC <- fsrfan(cycles~., data=wool, nsamp=0)

##  Store the score test statistic using Yeo and Johnson transformation.
outYJ <- fsrfan(cycles~., data=wool, family="YJ", nsamp=0)

## Not yet fully implemented
##  fanplot(outBC, main="Box Cox")
##  fanplot(outYJ,main="Yeo and Johnson")

plot(outBC, main="Box Cox")
plot(outYJ, main="Yeo and Johnson")

cat("\nMaximum difference in absolute value: ",
    max(max(abs(outYJ$Score - outBC$Score), na.rm=TRUE)), "\n")



##======================
  ## Call 'fsrfan' with Yeo-Johnson (YJ) transformation
  out <- fsrfan(cycles~., data=wool, family="YJ")
  plot(out)


## End(Not run)
```

`fsrfan.object` *Objects returned by the function* `fsrfan`

Description

An object of class `fsrfan.object` holds information about the result of a call to `fsrfan`.

Value

The functions `print()` and `summary()` are used to obtain and print a summary of the results. An object of class `fsrfan` is a list containing at least the following components:

1. `la` vector containing the values of lambda for which fan plot is constructed
2. `bs` matrix of size `p` X `length(la)` containing the units forming the initial subset for each value of lambda
3. `Score` a matrix containing the values of the score test for each value of the transformation parameter:
   - 1st col = fwd search index;
   - 2nd col = value of the score test in each step of the fwd search for la[1]

- ...

4. Scorep matrix containing the values of the score test for positive observations for each value of the transformation parameter.

   Note: this output is present only if input option family='YJpn' or family='YJall'.

5. Scoren matrix containing the values of the score test for negative observations for each value of the transformation parameter.

   Note: this output is present only if input option 'family' is 'YJpn' or 'YJall'.

6. Scoreb matrix containing the values of the score test for the joint presence of both constructed variables (associated with positive and negative observations) for each value of the transformation parameter. In this case the reference distribution is the F with 2 and subset_size - p degrees of freedom.

   Note: this output is present only if input option family='YJall'.

7. Un a three-dimensional array containing length(la) matrices of size retnUn=(n-init) X retpUn=11. Each matrix contains the unit(s) included in the subset at each step in the search associated with the corresponding element of la.

   REMARK: at each step the new subset is compared with the old subset. Un contains the unit(s) present in the new subset but not in the old one.

Examples

```
## Not run:
  data(wool)
  XX <- wool
  y <- XX[, ncol(XX)]
  X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

  out <- fsrfan(X, y)

  class(out)
  summary(out)

## End(Not run)
```

---

geyser2 *Old Faithful Geyser Data.*

Description

A bivariate data set obtained from the Old Faithful Geyser, containing the eruption length and the length of the previous eruption for 271 eruptions of this geyser in minutes.

Usage

```
data(geyser2)
```

Format

A data frame with 271 rows and 2 variables The variables are as follows:

- Eruption length: The eruption length in minutes.
- Previous eruption length: The length of the previous eruption in minutes.

References

Garcia-Escudero, L.A., Gordaliza, A. (1999). Robustness properties of k-means and trimmed k-means, *Journal of the American Statistical Assoc.*, Vol.**94**, No.447, 956-969.

Haerdle, W. (1991). *Smoothing Techniques with Implementation in S*, New York: Springer.

---

hawkins *Hawkins data.*

Description

These data, simulated by Hawkins, consist of 128 observations and eight explanatory variables (`X1`, `...`, `X8`) and one dependent variable, `y`.

Usage

```
data(hawkins)
```

Format

A data frame with 128 rows and 9 variables

---

hospital *Hospital data (Neter et al., 1996)*

Description

Data on the logged survival time of 108 patients undergoing liver surgery, together with four potential explanatory variables. Data are composed of 54 observations plus other 54 observations, introduced to check the model fitted to the first 54. Their comparison suggests there is no systematic difference between the two sets. However by looking at some FS plots (Riani and Atkinson, 2007), we conclude that these two groups are significantly different

Usage

```
data(hospital)
```

Format

A data frame with 108 rows and 5 variables The variables are as follows:

- X1
- X2
- X3
- X4
- y

@source J. NETER, M. H. KUTNER, C. J. NACHTSHEIM, W.WASSERMAN, *Applied Linear Statistical Models* (4th edition). McGraw-Hill, New York, 1996.

@references A. C. ATKINSON, M. RIANI, *Robust Diagnostic Regression Analysis*. Springer-Verlag, New York, 2000.

`Income1` *Income1*

Description

Income data taken from the United States Census Bureau. The data are a random sample of 200 observations referred to four variables. The goal is to predict HTOTVAL.

Usage

```
data(Income1)
```

Format

A data frame with 200 rows and 4 variables. The variables are as follows:

- H_NUMPER: Number of persons in household
- HOTHVAL: All other types of income except HEARNVAL Recode - Total other household income
- HSSVAL: household income - social security
- HTOTVAL: total household income (dollar amount)

Source

United States Census Bureau (2021). Annual Social and Economic Supplements

Examples

```
data(Income1)
head(Income1)
```

Income2 *Income2*

Description

A sample of 200 observations of full time employees from a municipality in Northern Italy who have declared extra income from investment sources. The variables are as follows. The goal is the possibility in predicting income level based on the individual's personal information.

Usage

```
data(Income2)
```

Format

A data frame with 200 rows and 6 variables. The variables are as follows:

- Age: Age of the person (the minimum is 19 and the maximum is 73).
- Education: Number of years of education (the minimum value of 5 is primary school, and the maximum value is 16 bachelor degree)
- Gender: A factor - Male or Female
- ExtraGain: Income from investment sources (profit-losses) apart from wages/salary
- Hours: total number of declared hours worked during the week. The minimum value is 35 and the maximum is 99
- Income: total yearly income (Euro amount)

Examples

```
data(Income2)
head(Income2)
```

levfwdplot *Plots the trajectories of the monitored scaled (squared) residuals*

Description

Plots the trajectories of the monitored scaled (squared) residuals

Usage

```
levfwdplot(out,
    xlim, ylim, xlab, ylab, main, lwd, lty, col, cex.lab, cex.axis,
    xvalues,
    fg.thresh, fg.unit, fg.labstep, fg.lwd, fg.lty, fg.col, fg.mark, fg.cex,
    bg.thresh, bg.style,
   xground=c("lev", "res"), tag, datatooltip, label, nameX, namey, msg, databrush,
    standard, fground, bground, ...)
```

Arguments

out	An object containing monitoring of leverage, fsreda.object. The needed elements of out are 1. LEV: matrix containing the leverage monitored in each step of the forward search. Every row is associated with a unit. This matrix can be created using function fsreg() with method="FS", monitoring=TRUE. 2. Un: (for FSR only) - matrix containing the order of entry in the subset of each unit (required only when datatooltip is true or databrush is not empty). 3. y: a vector containing the response (required only when option databrush is requested). 4. X: a matrix containing the explanatory variables (required only when option databrush is requested). 5. Bols: (n-init+1) x (p+1) matrix containing the estimated beta coefficients monitored in each step of the robust procedure (required only when option databrush is requested and suboption multivarfit is requested).
ylim	Control y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
xlim	Control x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
lwd	The line width, a positive number, defaulting to 1
lty	The line type. Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.
col	colors to be used for the highlighted units
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
xvalues	values for the x axis. Numeric vector of ncol(RES) controlling the x axis coordinates. The default value of xvalues is (nrow(RES) - ncol(RES) + 1):nrow(RES)
fg.thresh	(alternative to fg.unit) numeric vector of length 1 or 2 which specifies the highlighted trajectories. If length(fthresh) == 1 the highlighted trajectories are those of units that throughtout the search had at leat once a residual greater (in absolute value) than thresh. The default value is fg.thresh=2.5. If length(fthresh) == 2 the highlighted trajectories are those of units that throughtout the search had a residual at leat once bigger than fg.thresh[2] or smaller than fg.thresh[1].
fg.unit	(alternative to fg.thresh), vector containing the list of the units to be highlighted. If fg.unit is supplied, fg.thresh is ignored.

fg.labstep — numeric vector which specifies the steps of the search where to put labels for the highlighted trajectories (units). The default is to put the labels at the initial and final steps of the search. flabstep='' means no label.

fg.lwd — The line width for the highlighted trajectories (units). Default is 1.

fg.lty — The line type for the highlighted trajectories (units). Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.

fg.col — colors to be used for the highlighted units.

fg.mark — Controlls whether to plot highlighted trajectories as symbols. if fg.mark==TRUE each line is plotted using a different symbol else no marker is used (default).

fg.cex — controls the font size of the labels of the trajectories in foreground.

bg.thresh — numeric vector of length 1 or 2 which specifies how to define the unimmportant trajectories. Unimmportant trajectories will be plotted using a colormap, in greysh or will be hidden. If length(thresh) == 1 the irrelevant units are those which always had a residual smaller (in absolute value) than thresh. If length(bthresh) == 2 the irrelevant units are those which always had a residual greater than bthresh(1) and smaller than bthresh(2). The default is: bg.thresh=2.5 if n > 100 and bg.thresh=-Inf if n <= 100 i.e. to treat all trajectories as important if n <= 100 and, if n > 100, to reduce emphasis only to trajectories having in all steps of the search a value of scaled residual smaller than 2.5.

bg.style — specifies how to plot the unimportant trajectories as defined in option bthresh.

1. bg.style="faint": unimportant trajectories are plotted using a colormap.
2. bg.style="hide": unimportant trajectories are hidden.
3. bg.style="greyish": unimportant trajectories are displayed in a faint grey.

When n>100 the default option is bg.style='faint'. When n <= 100 and bg.thresh == -Inf option bstyle is ignored. Remark: bground=" is equivalent to -Inf that is all trajectories are considered relevant.

tag — Plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_resfwd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

xground — trajectories to highlight in connection with resfwdplot. If xground="lev" (default), the levfwdplot trajectories are put in foreground or in background depending on the leverage values. If xground="res", the levfwdplot trajectories are put in foreground or in background depending on the residual values. See options bg.thresh and fg.thresh.

datatooltip — Interactive clicking. It is inactive if this parameter is missing or empty. The default is datatooltip=TRUE, i.e. the user can select with the mouse an individual residual trajectory in order to have information about the corresponding unit. The information displayed depends on the estimator in use.

For example for class fsreda.object the information concerns the label and the step of the search in which the unit enters the subset. If datatooltip is a list it may contain the following fields:

1. DisplayStyle determines how the data cursor displays. Possible values are 'datatip' and 'window' (default). 'datatip' displays data cursor information in a small yellow text box attached to a black square marker at a data point you interactively select. 'window' displays data cursor information for the data point you interactively select in a floating window within the figure.
2. SnapToDataVertex: specifies whether the data cursor snaps to the nearest data value or is located at the actual pointer position. Possible values are SnapToDataVertex='on' (default) and SnapToDataVertex='off'.
3. LineColor: controls the color of the trajectory selected with the mouse. It can be an RGB triplet of values between 0 and 1, or character vector indicating a color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation.
4. SubsetLinesColor: enables to control the color of the trajectories of the units that are in the subset at a given step of the search (if levfwdplot() is applied to an object of class fsreda.object) or have a weight greater than 0.9 (if levfwdplot() is applied to an object of class sregeda.object or mmregeda.object). This can be done (repeatedly) with a left mouse click in proximity of the step of interest. A right mouse click will terminate the selection by marking with a up-arrow the step corresponding to the highlighted lines. The highlighted lines by default are in red, but a different color can be specified as RGB triplet or character of color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation. By default SubsetLinesColor="", i.e. the modality is not active. Any initialization for SubsetLinesColor which cannot be interpreted as RGB vector will be converted to blue, i.e. SubsetLinesColor will be forced to be [0 0 1]. If SubsetLinesColor is not empty the previous option LineColor is ignored.

label — Character vector containing the labels of the units (optional argument used when datatooltip=TRUE. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

nameX — Add variable labels in plot. A vector of strings of length p containing the labels of the variables of the regression dataset. If it is empty (default) the sequence X1, ..., Xp will be created automatically

namey — Add response label. A string containing the label of the response

msg — Controls whether to display or not messages on the screen If msg==1 (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

databrush — interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is a scalar or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the y|X plot, i.e. a matrix of scatter plots of y against each column of X, grouped according to the selection(s) done by brushing. If the plot y|X

does not exist it is automatically created. In addition, brushed units are automatically highlighted in the minimum deletion residual plot if it is already open. The extension to the following plots will be available in future versions of the toolbox:

1. monitoring leverage plot;
2. maximum studentized residual;
3. s^2 and R^2;
4. Cook distance and modified Cook distance;
5. deletion t statistics.

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of function `selectdataFS()` and the following optional argument:

1. `persist`. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
2. `bivarfit`. Wheather to superimpose bivariate least square lines on the plot (if `plot=TRUE`. This option adds one or more least squares lines, based on SIMPLE REGRESSION of y on Xi, to the plots of y|Xi. The default is `bivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. If `bivarfit=2`, two OLS lines are fitted: one to all points and another to the group of the genuine observations. The group of the potential outliers is not fitted. If `bivarfit=0` one OLS line is fitted to each group. This is useful for the purpose of fitting mixtures of regression lines. If `bivarfit='i1'` or `bivarfit='i2'`, etc. an OLS line is fitted to a specific group, the one with index 'i' equal to 1, 2, 3 etc. Again, useful in case of mixtures.
3. `multivarfit`. Wheather to superimpose multivariate least square lines. This option adds one or more least square lines, based on MULTIVARIATE REGRESSION of y on X, to the plots of y|Xi. The default is `multivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. The line added to the scatter plot y|Xi is avconst + Ci*Xi, where Ci is the coefficient of Xi in the multivariate regression and avconst is the effect of all the other explanatory variables different from Xi evaluated at their centroid (that is overline(y)'C)). If `multivarfit=2`, same action as with `multivarfit=1` but this time we also add the line based on the group of unselected observations (i.e. the normal units).

4. labeladd. Add outlier labels in plot. If labeladd=TRUE, we label the outliers with the unit row index in matrices X and y. The default value is labeladd=FALSE, i.e. no label is added.

standard	(MATLAB-style arguments) appearance of the plot in terms of xlim, ylim, axes labels and their font size style, color of the lines, etc.
fground	MATLAB-style arguments for the fground trajectories in foregroud.
bground	MATLAB-style arguments for the fground trajectories in backgroud.
...	potential further arguments passed to lower level functions.

Details

No details

Value

No value returned

Author(s)

FSDA team

Examples

```
## Not run:

n <- 100
y <- rnorm(n)
X <- matrix(rnorm(n*4), nrow=n)

out <- fsreg(y~X, method="LTS")
out <- fsreg(y~X, method="FS", bsb=out$bs, monitoring=TRUE)
levfwdplot(out)


## End(Not run)
```

---

loyalty *Loyalty data*

Description

The loyalty data consist of 509 observations on the behaviour of customers with loyalty cards from a supermarket chain in Northern Italy. The response y is the amount in euros spent at the shop over six months and the explanatory variables are: X1, the number of visits to the supermarket in the six month period; X2, the age of the customer; X3, the number of members of the customers' family. To find out more about this data set please see Atkinson and Riani (2006), JCGS

Usage

```
data("loyalty")
```

Format

A data frame with 509 observations on the following 4 variables.

`visits` the number of visits to the supermarket in the six month period

`age` the age of the customer

`family` the number of members of the customers' family

`amount_spent` the amount in euros spent at the shop over six months

Details

To find out more about this data set please see Atkinson and Riani (2006), JCGS

Source

The data are themselves a random sample from a larger database.

References

Atkinson, A. and Riani, M (2006) Distribution Theory and Simulations for Tests of Outliers in Regression, *Journal of Computational and Graphical Statistics*, **15** 2, pp 460–476.

Examples

```
data(loyalty)
```

LXS_control *Creates an* LSX_control *object*

Description

Creates an object of class `LXS_control` to be used with the `fsreg()` function, containing various control parameters.

Usage

```
LXS_control(intercept = TRUE, lms, h, bdp, nsamp, rew = FALSE, conflev = 0,
    msg = TRUE, nocheck = FALSE, nomes = FALSE, plot = FALSE)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
lms	Criterion to use to find the initial subset to initialize the search (LMS, LTS with concentration steps, LTS without concentration steps or subset supplied directly by the user). The default value is 1 (Least Median of Squares is computed to initialize the search). On the other hand, if the user wants to initialze the search with LTS with all the default options for concentration steps then lms=2. If the user wants to use LTS without concentration steps, lms can be a scalar different from 1 or 2. If lms is a list it is possible to control a series of options for concentration steps (for more details see option lms inside LXS_control). If, on the other hand, the user wants to initialize the search with a prespecified set of units there are two possibilities: 1. lms can be a vector with length greater than 1 which contains the list of units forming the initial subset. For example, if the user wants to initialize the search with units 4, 6 and 10 then lms=c(4, 6, 10); 2. lms is a struct which contains a field named bsb which contains the list of units to initialize the search. For example, in the case of simple regression through the origin with just one explanatory variable, if the user wants to initialize the search with unit 3 then lms=list(bsb=3).
h	The number of observations that have determined the least trimmed squares estimator, scalar. h is an integer greater or equal than p but smaller then n. Generally if the purpose is outlier detection h=[0.5*(n+p+1)] (default value). h can be smaller than this threshold if the purpose is to find subgroups of homogeneous observations. In this function the LTS/LMS estimator is used just to initialize the search.
bdp	Breakdown point. It measures the fraction of outliers the algorithm should resist. In this case any value greater than 0 but smaller or equal than 0.5 will do fine. If on the other hand the purpose is subgroups detection then bdp can be greater than 0.5. In any case however n*(1-bdp) must be greater than p. If this condition is not fulfilled an error will be given. Please specify h or bdp not both.
nsamp	Number of subsamples which will be extracted to find the robust estimator, scalar. If nsamp=0 all subsets will be extracted. They will be (n choose p). If the number of all possible subset is <1000 the default is to extract all subsets otherwise just 1000.
rew	LXS reweighted - if rew=1 the reweighted version of LTS (LMS) is used and the output quantities refer to the reweighted version else no reweighting is performed (default).
conflev	Confidence level which is used to declare units as outliers, usually conflev=0.95, 0.975, 0.99 (individual alpha) or 1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975.
msg	Controls whether to display or not messages on the screen If msg==1 (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

nocheck	Check input arguments, scalar. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. As default nocheck=FALSE.
nomes	It controls whether to display or not on the screen messages about estimated time to compute LMS (LTS). If nomes is equal to 1 no message about estimated time to compute LMS (LTS) is displayed, else if nomes is equal to 0 (default), a message about estimated time is displayed.
plot	Plot on the screen. Scalar. If plots=TRUE the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If plots=2 the user can also monitor the intermediate plots based on envelope superimposition. If plots=FALSE (default) no plot is produced.

Details

Creates an object of class FSR_control to be used with the fsreg() function, containing various control parameters.

Value

An object of class "LXS_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as Sreg_control, MMreg_control and FSR_control

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="LMS", control=LXS_control(h=56, nsamp=500, lms=2)))

## End(Not run)
```

M5data *Mixture M5 Data.*

Description

A bivariate data set obtained from three normal bivariate distributions with different scales and proportions 1:2:2. One of the components is strongly overlapping with another one. A 10 noise is added uniformly distributed in a rectangle containing the three normal components and not strongly overlapping with the three mixture components. A precise description of the M5 data set can be found in Garcia-Escudero et al. (2008).

Usage

```
data(M5data)
```

Format

A data frame with 2000 rows and 3 variables The first two columns are the two variables. The last column is the true classification vector where symbol "0" stands for the contaminating data points.

Source

Garcia-Escudero, L.A., Gordaliza, A., Matran, C. and Mayo-Iscar, A. (2008). A General Trimming Approach to Robust Cluster Analysis, *Annals of Statistics*, Vol.**36**, 1324-1345. doi:10.1214/07-AOS515.

`malfwdplot` *Plots the trajectories of scaled Mahalanobis distances along the search*

Description

Plots the trajectories of scaled Mahalanobis distances along the forward search

Usage

```
malfwdplot(
  out,
  xlim,
  ylim,
  xlab,
  ylab,
  main,
  lwd,
  lty,
  col,
  cex.lab,
  cex.axis,
  subsize,
  fg.thresh,
  fg.unit,
  fg.labstep,
  fg.lwd,
  fg.lty,
  fg.col,
  fg.mark,
  fg.cex,
  bg.thresh,
  bg.style,
```

```
  standard,
  fground,
  bground,
  tag,
  datatooltip,
  label,
  nameX,
  databrush,
  conflev,
  trace = FALSE,
  ...
)
```

Arguments

out	An object of S3 class fsmeda.object returned by fsmult with monitoring=TRUE - a list containing the monitoring of minimum Mahalanobis distance. The needed elements of out are 1. MAL: matrix containing the squared Mahalanobis distances monitored in each step of the forward search. Every row is associated with a unit (row of data matrix X). 2. Un: matrix containing the order of entry of each unit (necessary if datatooltip or databrush is selected). 3. X: The data matrix.
xlim	Controls the x scale in the plot. xlim is a vector with two elements controlling minimum and maximum on the x-axis. Default is to use automatic scale.
ylim	Controls the y scale in the plot. ylim is a vector with two elements controlling minimum and maximum on the y-axis. Default is to use automatic scale.
xlab	A title for the x axis
ylab	A title for the y axis, deafult is "Squared Mahalanobis distances".
main	An overall title for the plot
lwd	The line width, a positive number, defaulting to 1
lty	The line type. Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.
col	Colors to be used for the highlighted units
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
subsize	Numeric vector containing the subset size with length equal to the number of columns of matrix of mahalanobis distances. The default value of subsize is (nrow(MAL) - ncol(MAL) + 1):nrow(MAL)

`fg.thresh` (alternative to fg.unit) numeric vector of length 1 or 2 which specifies the highlighted trajectories. If `length(fg.thresh) == 1` the highlighted trajectories are those of units that throughtout the search had at leat once a mahalanobis distance greater than `fg.thresh`. The default value is `fg.thresh=2.5`. If `length(fg.thresh) == 2` the highlighted trajectories are those of units that throughtout the search had a mahalanobis distance at least once bigger than `fg.thresh[2]` or smaller than `fg.thresh[1]`.

`fg.unit` (alternative to fg.thresh), vector containing the list of the units to be highlighted. If `fg.unit` is supplied, `fg.thresh` is ignored.

`fg.labstep` numeric vector which specifies the steps of the search where to put labels for the highlighted trajectories (units). The default is to put the labels at the initial and final steps of the search. `fg.labstep=''` means no label.

`fg.lwd` The line width for the highlighted trajectories (units). Default is 1.

`fg.lty` The line type for the highlighted trajectories (units). Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.

`fg.col` colors to be used for the highlighted units.

`fg.mark` Controlls whether to plot highlighted trajectories as symbols. if `fg.mark==TRUE` each line is plotted using a different symbol else no marker is used (default).

`fg.cex` Controls the font size of the labels of the trajectories in foreground. If `fg.cex=0` no labels will be shown - equivalent to `fg.labstop=""`.

`bg.thresh` Numeric vector of length 1 or 2 which specifies how to define the *unimmportant trajectories*. Unimmportant trajectories will be plotted using a colormap, in greysh or will be hidden. If `length(bg.thresh) == 1` the irrelevant units are those which always had a mahalanobis distance smaller than `bg.thresh`. If `length(bg.thresh) == 2` the irrelevant units are those which always had a mahalanobis distance greater than `bg.thresh[1]` and smaller than `bg.thresh[2]`. The default is `bg.thresh=2.5` if `n > 100` and `bg.thresh=-Inf` if `n <= 100` i.e. to treat all trajectories as important if `n <= 100` and, if `n > 100`, to reduce emphasis only to trajectories having in all steps of the search a value of mahalanobis distance smaller than 2.5.

`bg.style` Specifies how to plot the unimportant trajectories as defined in option bg.thresh.

1. `bg.style="faint"`: unimportant trajectories are plotted using a colormap.
2. `bg.style="hide"`: unimportant trajectories are hidden.
3. `bg.style="greyish"`: unimportant trajectories are displayed in a faint grey.

When `n > 100` the default option is `bg.style='faint'`. When `n <= 100` and `bg.thresh == -Inf` option bg.style is ignored. Remark: bground=" is equivalent to -Inf that is all trajectories are considered relevant.

`standard` MATLAB-style arguments - appearance of the plot in terms of xlim, ylim, axes labels and their font size style, color of the lines, etc.

`fground` MATLAB-style arguments - for the trajectories in foregroud.

`bground` MATLAB-style arguments - for the trajectories in backgroud.

`tag` Plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_resfwd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

`datatooltip` Interactive clicking. It is inactive if this parameter is set to FALSE. The default is `datatooltip=TRUE`, the user can select with the mouse an individual mahalanobis distance trajectory in order to have information about the corresponding unit, the associated label and the step of the search in which the unit enters the subset. If datatooltip is a list it may contain the following fields:

1. DisplayStyle determines how the data cursor displays. Possible values are `'datatip'` and `'window'` (default). `'datatip'` displays data cursor information in a small yellow text box attached to a black square marker at a data point you interactively select. `'window'` displays data cursor information for the data point you interactively select in a floating window within the figure.
2. SnapToDataVertex: specifies whether the data cursor snaps to the nearest data value or is located at the actual pointer position. Possible values are `SnapToDataVertex='on'` (default) and `SnapToDataVertex='off'`.
3. LineColor: controls the color of the trajectory selected with the mouse. It can be an RGB triplet of values between 0 and 1, or character vector indicating a color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation.
4. SubsetLinesColor: enables to control the color of the trajectories of the units that are in the subset at a given step of the search (if `resfwdplot()` is applied to an object of class `fsreda.object`) or have a weight greater than 0.9 (if `resfwdplot()` is applied to an object of class `sregeda.object` or `mmregeda.object`). This can be done (repeatedly) with a left mouse click in proximity of the step of interest. A right mouse click will terminate the selection by marking with a up-arrow the step corresponding to the highlighted lines. The highlighted lines by default are in red, but a different color can be specified as RGB triplet or character of color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation. By default `SubsetLinesColor=""`, i.e. the modality is not active. Any initialization for `SubsetLinesColor` which cannot be interpreted as RGB vector will be converted to blue, i.e. `SubsetLinesColor` will be forced to be [0 0 1]. If `SubsetLinesColor` is not empty the previous option `LineColor` is ignored.

`label` Character vector containing the labels of the units (optional argument used when `datatooltip=TRUE`. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

`nameX` Add variable labels in plot. A vector of strings of length `p` containing the labels of the variables in the dataset. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

`databrush` Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is `TRUE` or a list) enables the

user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. In addition, brushed units can be highlighted in the monitoring mahalanobis distance plot. Note that the window style of the other figures is set equal to that which contains the monitoring mahalanobis distance plot. In other words, if the monitoring mahalanobis distance plot is docked all the other figures will be docked too.

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of the MATLAB function `selectdataFS()` and the following optional arguments:

- persist. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
- `label`: add labels of brushed units in the monitoring plot.
- `labeladd`: add labels of brushed units in the scatterplot matrix. If this option is '1', we label the units of the last selected group with the unit row index in the matrix X. The default value is labeladd='', i.e. no label is added.

`conflev` confidence interval for the horizontal bands. It can be a vector of different confidence level values, e.g. c(0.95, 0.99, 0.999). The confidence interval is based on the chi^2 distribution.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson A.C., Riani M. and Cerioli A. (2004), Exploring Multivariate Data with the Forward Search, Springer Verlag, New York.

Examples

```
## Not run:
## Produce monitoring MD plot with all the default options.
##  Generate input structure for malfwdplot
n <- 100
p <- 4
```

```
Y <- matrix(rnorm(n*p), ncol=p)
Y[1:10,] <- Y[1:10,] + 4

out <- fsmult(Y, monitoring=TRUE, init=30)

##  Produce monitoring MD plot with all the default options
malfwdplot(out)

## End(Not run)
```

`malindexplot` *Plots the trajectory of minimum Mahalanobis distance (mmd)*

Description

Plots the trajectory of minimum Mahalanobis distance (mmd)

Usage

```
malindexplot(
  out,
  p,
  xlab,
  ylab,
  main,
  nameX,
  conflev,
  numlab,
  tag,
  trace = FALSE,
  ...
)
```

Arguments

`out`	a numeric vector or an object of S3 class (one of `fsmult.object`, `smult.object` or `mmmult.object`) returned by one of the functions `fsmult` or `smult` or `mmmult` - a list containing the monitoring of minimum Mahalanobis distance
`p`	If `out` is a vector, p is the number of variables of the original data matrix which have been used to compute md.
`xlab`	A title for the x axis
`ylab`	A title for the y axis
`main`	An overall title for the plot
`nameX`	Add variable labels in the plot. A vector of strings of length `p` containing the labels of the variables of the original data matrix `X`. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

conflev	confidence interval for the horizontal bands. It can be a vector of different confidence level values, e.g. c(0.95, 0.99, 0.999). The confidence interval is based on the chi^2 distribution.
numlab	Number of points to be labeled in the plot. If numlab is a single number, e.g. numlab]k, the units with the k largest md are labelled in the plots. If numlab is a vector, the units indexed by the vector are labelled in the plot. Default is numlab=5, i.e. the 5 units units with the largest md are labelled. Use numlab="" for no labelling.
tag	Tag of the figure which will host the malindexplot. The default tag is tag="pl_malindex".
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson and Riani (2000), Robust Diagnostic Regression Analysis, Springer Verlag, New York.

Examples

```
## Not run:
##  Mahalanobis distance plot of 100 random numbers.
##  Numbers are from from the chisq with 5 degrees of freedom

malindexplot(rchisq(100, 5), 5)

## End(Not run)
```

---

mdrplot *Plots the trajectory of minimum deletion residual (mdr)*

Description

Plots the trajectory of minimum deletion residual (mdr).

Usage

```
mdrplot(out, quant = c(0.01, 0.5, 0.99), sign = TRUE,
    mplus1 = FALSE, envm,
    xlim, ylim, xlab, ylab, main,
    lwdenv, lwd, cex.lab, cex.axis,
    tag, datatooltip, label, nameX, namey, databrush,
    ...)
```

Arguments

out	An object returned by FSReda() (see FSReda_control). The needed elements of out are 1. mdr: Minimum deletion residual. A matrix containing the monitoring of minimum deletion residual in each step of the forward search. The first column of mdr must contain the fwd search index. 2. Un: (for FSR only) - matrix containing the order of entry in the subset of each unit (required only when datatooltip is true or databrush is not empty). 3. y: a vector containing the response (required only when option databrush is requested). 4. X: a matrix containing the explanatory variables (required only when option databrush is requested). 5. Bols: (n-init+1) x (p+1) matrix containing the estimated beta coefficients monitored in each step of the robust procedure (required only when option databrush is requested and suboption multivarfit is requested).
quant	Quantiles for which envelopes have to be computed. The default is to produce 1%, 50% and 99% envelopes. In other words the default is quant=c(0.01, 0.5, 0.99).
sign	Wheather to use MDR with sign: if sign=TRUE (default) we distinguish steps for which minimum deletion residual was associated with positive or negative value of the residual. Steps associated with positive values of mdr are plotted in black, while other steps are plotted in red.
mplus1	Wheather to plot the (m+1)-th order statistic. Specifies if it is necessary to plot the curve associated with (m+1)-th order statistic.
envm	Sample size for drawing enevlopes. Specifies the size of the sample which is used to superimpose the envelope. The default is to add an envelope based on all the observations (size n envelope).
ylim	Control y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
xlim	Control x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
lwdenv	Controls the width of the lines associated with the envelopes, default is lvdenv=1.

`lwd` Controls the linewidth of the curve which contains the monitoring of minimum deletion residual.

`cex.lab` The magnification to be used for x and y labels relative to the current setting of cex

`cex.axis` The magnification to be used for axis annotation relative to the current setting of cex

`tag` Plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_mdr'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

`datatooltip` If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function `datacursormode()` for more details or see the examples below). The default options are `DisplayStyle="Window"` and `SnapToDataVertex="on"`.

`label` Character vector containing the labels of the units (optional argument used when `datatooltip=TRUE`. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

`nameX` Add variable labels in plot. A vector of strings of length `p` containing the labels of the variables of the regression dataset. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

`namey` Add response label. A string containing the label of the response

`databrush` interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is a scalar or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the y|X plot, i.e. a matrix of scatter plots of y against each column of X, grouped according to the selection(s) done by brushing. If the plot y|X does not exist it is automatically created. In addition, brushed units are automatically highlighted in the minimum deletion residual plot if it is already open. The extension to the following plots will be available in future versions of the toolbox:

1. monitoring leverage plot;
2. maximum studentized residual;
3. s^2 and R^2;
4. Cook distance and modified Cook distance;
5. deletion t statistics.

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of function `selectdataFS()` and the following optional argument:

1. `persist`. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
2. `bivarfit`. Wheather to superimpose bivariate least square lines on the plot (if `plot=TRUE`. This option adds one or more least squares lines, based on SIMPLE REGRESSION of y on Xi, to the plots of y|Xi. The default is `bivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. If `bivarfit=2`, two OLS lines are fitted: one to all points and another to the group of the genuine observations. The group of the potential outliers is not fitted. If `bivarfit=0` one OLS line is fitted to each group. This is useful for the purpose of fitting mixtures of regression lines. If `bivarfit='i1'` or `bivarfit='i2'`, etc. an OLS line is fitted to a specific group, the one with index 'i' equal to 1, 2, 3 etc. Again, useful in case of mixtures.
3. `multivarfit`. Wheather to superimpose multivariate least square lines. This option adds one or more least square lines, based on MULTIVARIATE REGRESSION of y on X, to the plots of y|Xi. The default is `multivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. The line added to the scatter plot y|Xi is avconst + Ci*Xi, where Ci is the coefficient of Xi in the multivariate regression and avconst is the effect of all the other explanatory variables different from Xi evaluated at their centroid (that is overline(y)'C)). If `multivarfit=2`, same action as with `multivarfit=1` but this time we also add the line based on the group of unselected observations (i.e. the normal units).
4. `labeladd`. Add outlier labels in plot. If `labeladd=TRUE`, we label the outliers with the unit row index in matrices X and y. The default value is `labeladd=FALSE`, i.e. no label is added.

`...` potential further arguments passed to lower level functions.

Details

No details

Value

No value returned

Author(s)

FSDA team

Examples

```
## Not run:
```

```
n <- 100
y <- rnorm(n)
X <- matrix(rnorm(n*4), nrow=n)

out <- fsreg(y~X, method="LTS")
out <- fsreg(y~X, method="FS", bsb=out$bs, monitoring=TRUE)
mdrplot(out)

## End(Not run)
```

`mmdplot` *Plots the trajectory of minimum Mahalanobis distance (mmd)*

Description

Plots the trajectory of minimum Mahalanobis distance (mmd)

Usage

```
mmdplot(
  out,
  quant = c(0.01, 0.5, 0.99),
  mplus1 = FALSE,
  envm,
  lwd,
  lwdenv,
  xlim,
  ylim,
  tag,
  datatooltip,
  label,
  xlab,
  ylab,
  main,
  nameX,
  cex.lab,
  cex.axis,
  databrush,
  trace = FALSE,
  ...
)
```

Arguments

`out`	An object of S3 class `fsmeda.object` returned by `fsmult` with `monitoring=TRUE` - a list containing the monitoring of minimum Mahalanobis distance

quant	Quantiles for which envelopes have to be computed. The default is to produce 1%, 50% and 99% envelopes. In other words the default is quant=c(0.01, 0.5, 0.99).
mplus1	Wheather to plot the (m+1)-th order statistic.
envm	Sample size for drawing enevlopes. Specifies the size of the sample which is used to superimpose the envelope. The default is to add an envelope based on all the observations (size n envelope).
lwd	Controls the line width of the curve which contains the monitoring of minimum deletion residual.
lwdenv	Controls the width of the lines associated with the envelopes. Default is lwdenv=1
xlim	Control the x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.
ylim	Control the y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
tag	Plot handle. String which identifies the handle of the plot which is about to be created. The default is tag='pl_mmd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.
datatooltip	If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function datacursormode() for more details or see the examples below). The default options are DisplayStyle="Window" and SnapToDataVertex="on".
label	Row labels. Character vector containing the labels of the units (optional argument used when datatooltip=TRUE. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).
xlab	A title for the x axis
ylab	A title for the y axis
main	An overall title for the plot
nameX	Add variable labels in the plot. A vector of strings of length p containing the labels of the variables of the original data matrix X. If it is empty (default) the sequence X1, ..., Xp will be created automatically
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
databrush	Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is TRUE or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is

automatically created. In addition, brushed units can be highlighted in the monitoring MD plot. Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too.

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of the MATLAB function `selectdataFS()` and the following optional arguments:

- `persist`: This option can be an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
- `labeladd`: add labels of brushed units in the scatterplot matrix. If this option is '1', we label the units of the last selected group with the unit row index in the matrix X. The default value is labeladd='', i.e. no label is added.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson and Riani (2000), Robust Diagnostic Regression Analysis, Springer Verlag, New York.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsmult(hbk[,1:3], monitoring=TRUE))
mmdplot(out)

## End(Not run)
```

`mmdrsplot` *Plots the trajectories of minimum Mahalanobis distances from different starting points*

Description

Plots the trajectories of minimum Mahalanobis distances from different starting points

Usage

```
mmdrsplot(
  out,
  quant = c(0.01, 0.5, 0.99),
  envm,
  lwd,
  lwdenv,
  xlim,
  ylim,
  tag,
  datatooltip,
  label,
  xlab,
  ylab,
  envlab = TRUE,
  main,
  nameX,
  cex.lab,
  cex.axis,
  databrush,
  scaled = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

`out` An object of S3 class `fsmmmdrs.object` returned by `fsmmmdrs` - a list containing the following elements:

- `mmdrs` = a matrix of size (n-ninit)-by-(nsimul+1) containing the monitoring of minimum Mahalanobis distance in each step of the forward search for each of the nsimul random starts. The first column of mmdrs must contain the forward search index. This matrix can be created using function `fsmmmdrs`.
- `BBrs` = 3D array of size n-by-n-(init)-by-nsimul containing units forming subset for rach random start. This field is necessary if datatooltip is true or databrush is not empty.

- X = n-by-v matrix containing the original data matrix. This field is necessary if datatooltip is true or databrush is not empty.

quant — Quantiles for which envelopes have to be computed. The default is to produce 1%, 50% and 99% envelopes. In other words the default is quant=c(0.01, 0.5, 0.99).

envm — Sample size for drawing enevlopes. Specifies the size of the sample which is used to superimpose the envelope. The default is to add an envelope based on all the observations (size n envelope).

lwd — Controls the linewidth of the curve which contains the monitoring of minimum deletion residual.

lwdenv — line width: a scalar which controls the width of the lines associated with the envelopes. Default is lwdenv=1

xlim — Control the x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.

ylim — Control the y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.

tag — Plot handle. String which identifies the handle of the plot which is about to be created. The default is tag='pl_mmdrs'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

datatooltip — If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function datacursormode() for more details or see the examples below). The default options are DisplayStyle="Window" and SnapToDataVertex="on".

label — Row labels. Character vector containing the labels of the units (optional argument used when datatooltip=TRUE. If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

xlab — A title for the x axis

ylab — A title for the y axis

envlab — wheather to label the envelopes. If envlab is true (default) labels of the confidence envelopes which are used are added on the y axis.

main — An overall title for the plot

nameX — Add variable labels in the plot. A vector of strings of length p containing the labels of the variables of the original data matrix X. If it is empty (default) the sequence X1, ..., Xp will be created automatically

cex.lab — The magnification to be used for x and y labels relative to the current setting of cex

cex.axis — The magnification to be used for axis annotation relative to the current setting of cex

`databrush` Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is TRUE or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. In addition, brushed units can be highlighted in the monitoring MD plot. Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too.

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of the MATLAB function `selectdataFS()` and the following optional arguments:

1. persist. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.

`scaled` Wheather to use scaled or unscaled envelopes. If `scaled=TRUE` the envelopes are produced for scaled Mahalanobis distances (no consistency factor is applied) else the traditional consistency factor is applied. Default is `scaled=FALSE`

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C., Riani, M. and Cerioli, A. (2004), '*Exploring multivariate data with the forward search*, Springer Verlag, New York.

Examples

```
## Not run:
data(hbk, package="robustbase")
out <- fsmmmdrs(hbk[,1:3])
```

```
 mmdrsplot(out)

## End(Not run)
```

`mmmult` *Computes MM estimators in multivariate analysis with auxiliary S-scale*

Description

Computes MM estimators in multivariate analysis with auxiliary S-scale

Usage

```
mmmult(
  x,
  monitoring = FALSE,
  plot = FALSE,
  eff,
  conflev = 0.975,
  nocheck = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

`x`	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
`monitoring`	Wheather to perform monitoring of Mahalanobis distances and other specific quantities
`plot`	Plots the Mahalanobis distances against index number. If `plot=FALSE` (default) or `plot=0` no plot is produced. The confidence level used to draw the confidence bands for the MD is given by the input option conflev. If conflev is not specified a nominal 0.975 confidence interval will be used. If `plot=2` a scatter plot matrix with the outliers highlighted is produced. If plot is a list it may contain the following fields: • labeladd If `labeladd=1`, the outliers in the spm are labelled with the unit row index. The default value is `labeladd=""`, i.e. no label is added • nameY character vector containing the labels of the variables. As default value, the labels which are added are Y1, ...Yp.
`eff`	Defining the nominal efficiency (i.e. a number between 0.5 and 0.99). The default value is `eff=0.95`.

conflev	Confidence level which is used to declare units as outliers (scalar). Usually conflev=0.95, conflev=0.975 or conflev=0.99 (individual alpha) conflev=1-0.05/n, conflev=1-0.025/n or conflev=1-0.01/n (simultaneous alpha). Default value is convlev=0.975.
nocheck	It controls whether to perform checks on matrix Y. If nocheck=TRUE, no check is performed.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Details

This function follows the lines of MATLAB/R code developed during the years by many authors. For more details see http://www.econ.kuleuven.be/public/NDBAE06/programs/ and the R package CovMMest The core of these routines, e.g. the resampling approach, however, has been completely redesigned, with considerable increase of the computational performance.

Value

Depending on the input parameter monitoring, one of the following objects will be returned:

1. mmmult.object
2. mmmulteda.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Maronna, R.A., Martin D. and Yohai V.J. (2006), Robust Statistics, Theory and Methods, Wiley, New York.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- mmmult(hbk[,1:3]))
class(out)
summary(out)

##  Generate contaminated data (200,3)
n <- 200
p <- 3
set.seed(123456)
X <- matrix(rnorm(n*p), nrow=n)
Xcont <- X
Xcont[1:5, ] <- Xcont[1:5,] + 3

out1 <- mmmult(Xcont, trace=TRUE)           # no plots (plot defaults to FALSE)
names(out1)
```

```
## plot=TRUE - generates: (1) a plot of Mahalanobis distances against
##    index number. The confidence level used to draw the confidence bands for
##    the MD is given by the input option conflev. If conflev is
##    not specified a nominal 0.975 confidence interval will be used and
##    (2) a scatter plot matrix with the outliers highlighted.

(out1 <- mmmult(Xcont, trace=TRUE, plot=TRUE))

## plots is a list: the spm shows the labels of the outliers.
(out1 <- mmmult(Xcont, trace=TRUE, plot=list(labeladd="1")))

## plots is a list: the spm uses the variable names provided by 'nameY'.
(out1 <- mmmult(Xcont, trace=TRUE, plot=list(nameY=c("A", "B", "C"))))

## mmmult() with monitoring
(out2 <- mmmult(Xcont, monitoring=TRUE, trace=TRUE))
names(out2)

##  Forgery Swiss banknotes examples.

data(swissbanknotes)

(out1 <- mmmult(swissbanknotes[101:200,], plot=TRUE))

(out1 <- mmmult(swissbanknotes[101:200,], plot=list(labeladd="1")))

## End(Not run)
```

`mmmult.object` *Description of* `mmmult.object` *Objects*

Description

An object of class `mmmult.object` holds information about the result of a call to `mmmult`.

Value

The object itself is basically a `list` with the following components:

`loc`	p-by-1 vector containing MM estimate of location.
`shape`	p-by-p matrix with MM estimate of the shape matrix.
`cov`	matrix with MM estimate of the covariance matrix. Remark: `covariance = auxscale^2 * shape`.
`weights`	A vector containing the estimates of the weights.
`outliers`	A vector containing the list of the units declared as outliers using confidence level specified in input scalar `conflev`.
`Sloc`	A vector with S estimate of location.

Sshape	A matrix with S estimate of the shape matrix.
Scov	A matrix with S estimate of the covariance matrix.
auxscale	S estimate of the scale.
md	n-by-1 vector containing the estimates of the robust Mahalanobis distances (in squared units).
conflev	Confidence level that was used to declare outliers.
X	the data matrix X

The object has class "mmmult".

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- mmmult(hbk[,1:3]))
class(out)
summary(out)

## End(Not run)
```

---

mmmulteda.object *Description of* mmmulteda.object *Objects*

Description

An object of class mmmulteda.object holds information about the result of a call to mmmult with monitoring=TRUE.

Value

The object itself is basically a list with the following components:

Loc	length(eff)-by-p matrix containing MM estimate of location for each value of eff.
Shape	p-by-p-by-length(eff) 3D array containing robust estimate of the shape for each value of eff. Remark: det|shape|=1.
Scale	length(eff) vector containing robust estimate of the scale for each value of eff.
Cov	p-by-p-by-length(eff) 3D array containing robust estimate of covariance matrix for each value of eff. Note that scale(i)^2 * shape[,,i] = robust estimate of covariance matrix.
Bs	(p+1)-by-length(eff) matrix containing the units forming best subset for each value of eff.
MAL	n-by-length(eff) matrix containing the estimates of the robust Mahalanobis distances (in squared units) for each value of eff.

Outliers	n-by-length(eff) matrix containing flags for the outliers. Boolean matrix containing the list of the units declared as outliers for each value of eff using confidence level specified in input scalar conflev
Weights	n x length(eff) matrix containing the weights for each value of eff.
conflev	Confidence level that was used to declare outliers.
singsub	Number of subsets without full rank. Notice that singsub > 0.1*(number of subsamples) produces a warning.
eff	vector which contains the values of eff which have been used.
X	the data matrix X.

The object has class "mmmulteda".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- mmmult(hbk[,1:3], monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

---

mmreg.object *Description of mmreg Objects*

Description

An object of class mmreg.object holds information about the result of a call to fsreg with method="MM".

Value

The object itself is basically a list with the following components:

beta	p-by-1 vector containing the MM estimate of regression coefficients.
auxscale	scalar, S estimate of the scale (or supplied external estimate of scale, if option InitialEst is not empty).
residuals	residuals.
fittedvalues	fitted values.
weights	n x 1 vector. Weights assigned to each observation.
Sbeta	p x 1 vector containing S estimate of regression coefficients (or supplied initial external estimate of regression coefficients, if option InitialEst is not empty)
Ssingsub	Number of subsets without full rank in the S preliminary part. Notice that out.singsub > 0.1*(number of subsamples) produces a warning.
outliers	kx1 vector containing the list of the k units declared as outliers or NULL if the sample is homogeneous.

conflev	Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975
rhofunc	Specifies the rho function which has been used to weight the residuals. If a different rho function is specified for S and MM loop then insted of rhofunc we will have rhofuncS and rhofuncMM.
rhofuncparam	Vector which contains the additional parameters for the specified rho function which has been used. For hyperbolic rho function the value of k =sup CVC. For Hampel rho function the parameters a, b and c. If a different rho function is specified for S and MM loop then insted of rhofuncparam we will have rhofuncparamS and rhofuncparamMM.
X	the data matrix X
y	the response vector y

The object has class "mmreg".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="MM"))
    class(out)
    summary(out)

## End(Not run)
```

MMreg_control *Creates an* MMreg_control *object*

Description

Creates an object of class MMreg_control to be used with the fsreg() function, containing various control parameters for calling the MATLAB function MMreg().

Usage

```
MMreg_control(intercept = TRUE, InitialEst, eff, effshape,
    rhofunc = c("bisquare", "optimal", "hyperbolic", "hampel", "mdpd", "AS"),
    rhofuncparam, refsteps = 3, tol = 1e-07, conflev,
    nocheck = FALSE, Smsg = TRUE, plot = FALSE)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.

`InitialEst` Starting values of the MM-estimator, a list with the fiollowing elements: `loc`, a $p x 1$ vector, location vector estimate and `scale`, a scaler, estimate of the scale. If empty (default) the program will use S estimators. In this last case it is possible to specify the options given in function Sreg.

`eff` Scalar defining nominal efficiency (i.e. a number between 0.5 and 0.99). The default value is 0.95.

`effshape` Location or scale efficiency. If `effshape=1` efficiency refers to shape efficiency else (default) efficiency refers to location efficiency.

`rhofunc` Specifies the rho function which must be used to weight the residuals. Possible values are 'bisquare' 'optimal' 'hyperbolic' 'hampel'.

1. 'bisquare' uses Tukey's rho and psi functions. See TBrho and TBpsi.
2. 'optimal' uses optimal rho and psi functions. See OPTrho and OPTpsi.
3. 'hyperbolic' uses hyperbolic rho and psi functions. See HYPrho and HYPpsi.
4. 'hampel' uses Hampel rho and psi functions. See HArho and HApsi.

The default is 'bisquare'.

`rhofuncparam` Additional parameters for the specified rho function. For hyperbolic rho function it is possible to set up the value of k = sup CVC (the default value of k is 4.5).

For Hampel rho function it is possible to define parameters a, b and c (the default values are a=2, b=4, c=8)

`refsteps` Number of refining iterations in each subsample (default is `refsteps=3`). `refsteps = 0` means "raw-subsampling" without iterations.

`tol` Scalar controlling tolerance in the MM loop. The default value is `tol=1e-6`

`conflev` Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975

`nocheck` Check input arguments, scalar. If `nocheck=TRUE` no check is performed on matrix `y` and matrix `X`. Notice that `y` and `X` are left unchanged. In other words the additional column of ones for the intercept is not added. As default `nocheck=FALSE`.

`Smsg` Controls whether to display or not messages on the screen If `Smsg==TRUE` (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

`plot` Plot on the screen. Scalar. If `plots=TRUE` the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plots=2` the user can also monitor the intermediate plots based on envelope superimposition. If `plots=FALSE` (default) no plot is produced.

Details

Creates an object of class `MMreg_control` to be used with the `fsreg()` function, containing various control parameters.

Value

An object of class "MMreg_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as FSR_control, MMreg_control and LXS_control

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="MM", control=MMreg_control(eff=0.99, rhofunc="optimal")))

## End(Not run)
```

---

mmregeda.object *Description of* mmregeda *Objects*

Description

An object of class mmregeda.object holds information about the result of a call to fsreg when method="MM" and monitoring=TRUE.

Value

The object itself is basically a list with the following components:

auxscale	scalar, S estimate of the scale (or supplied external estimate of scale, if option InitialEst is not empty).
Beta	p x length(eff) matrix containing MM estimate of regression coefficients for each value of eff.
RES	n x length(eff) matrix containing the monitoring of scaled residuals for each value of eff.
Weights	n x length(eff) matrix containing the estimates of the weights for each value of eff
Outliers	Boolean matrix containing the list of the units declared as outliers for each value of eff using confidence level specified in input scalar conflev.
conflev	Confidence level which is used to declare units as outliers. Remark: conflev will be used to draw the horizontal line (confidence band) in the plot.
Ssingsub	Number of subsets wihtout full rank. Notice that Notice that singsub > 0.1*(number of subsamples) produces a warning

rhofunc	string identifying the rho function which has been used.
rhofuncparam	vector which contains the additional parameters for the specified rho function which have been used. For hyperbolic rho function the value of k =sup CVC. For Hampel rho function the parameters a, b and c.
eff	vector containing the value of eff which have been used.
X	the data matrix X
y	the response vector y

The object has class "mmregeda".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="MM", monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

MMregeda_control *Creates an* MMregeda_control *object*

Description

Creates an object of class MMregeda_control to be used with the fsreg() function, containing various control parameters.

Usage

```
MMregeda_control(intercept = TRUE, InitialEst, Soptions, eff, effshape,
 rhofunc = c("bisquare", "optimal", "hyperbolic", "hampel", "mdpd", "AS"),
rhofuncparam, refsteps = 3, tol = 1e-07, conflev, nocheck = FALSE, plot = FALSE)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
InitialEst	Starting values of the MM-estimator, a list with the fiollowing elements: loc, a $p x 1$ vector, location vector estimate and scale, a scaler, estimate of the scale. If empty (default) the program will use S estimators. In this last case it is possible to specify the options given in function Sreg.
Soptions	Options to pass to Sreg, an Sreg_control object. The options are: Srhofunc, Snsamp, Srefsteps, Sreftol, Srefstepsbestr, Sreftolbestr, Sminsctol, Sbestr. See function Sreg_control for more details on these options.

It is necessary to add to the S options the letter S at the beginning. For example, if you want to use the optimal rho function the supplied option is 'Srhofunc','optimal'. For example, if you want to use 3000 subsets, the supplied option is 'Snsamp',3000

`eff` Vector defining nominal efficiency (i.e. a series of numbers between 0.5 and 0.99). The default value is the sequence `seq(0.5, 0.99, 0.01)`

`effshape` Location or scale efficiency. If `effshape=1` efficiency refers to shape efficiency else (default) efficiency refers to location efficiency.

`rhofunc` Specifies the rho function which must be used to weight the residuals. Possible values are 'bisquare' 'optimal' 'hyperbolic' 'hampel'.

1. 'bisquare' uses Tukey's rho and psi functions. See TBrho and TBpsi.
2. 'optimal' uses optimal rho and psi functions. See OPTrho and OPTpsi.
3. 'hyperbolic' uses hyperbolic rho and psi functions. See HYPrho and HYPpsi.
4. 'hampel' uses Hampel rho and psi functions. See HArho and HApsi.

The default is 'bisquare'.

`rhofuncparam` Additional parameters for the specified rho function. For hyperbolic rho function it is possible to set up the value of k = sup CVC (the default value of k is 4.5).

For Hampel rho function it is possible to define parameters a, b and c (the default values are a=2, b=4, c=8)

`refsteps` Number of refining iterations in each subsample (default is `refsteps=3`). `refsteps = 0` means "raw-subsampling" without iterations.

`tol` Scalar controlling tolerance in the MM loop. The default value is `tol=1e-6`.

`conflev` Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975

`nocheck` Check input arguments, scalar. If `nocheck=TRUE` no check is performed on matrix `y` and matrix `X`. Notice that `y` and `X` are left unchanged. In other words the additional column of ones for the intercept is not added. As default `nocheck=FALSE`.

`plot` Plot on the screen. Scalar. If `plots=TRUE` the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plots=2` the user can also monitor the intermediate plots based on envelope superimposition. If `plots=FALSE` (default) no plot is produced.

Details

Creates an object of class `MMregeda_control` to be used with the `fsreg()` function, containing various control parameters.

Value

An object of class `"MMregeda_control"` which is basically a `list` with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as FSR_control, Sreg_control, MMreg_control and LXS_control

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="MM", monitoring=TRUE,
    control=MMregeda_control(eff=seq(0.75, 0.99, 0.01))))

## End(Not run)
```

---

multiple_regression — *Multiple regression data showing the effect of masking (Atkinson and Riani, 2000).*

Description

There are 60 observations on a response y with the values of three explanatory variables. The scatter plot matrix of the data shows y increasing with each of x1, x2 and x3. The plot of residuals against fitted values shows no obvious pattern. However the FS finds that there are 6 masked outliers.

Usage

```
data(multiple_regression)
```

Format

A data frame with 60 rows and 4 variables The variables are as follows:

- X1
- X2
- X3
- y

@references Atkinson, A. C., and Riani, M. (2000). *Robust Diagnostic Regression Analysis*. Springer-Verlag, New York.

mussels *Mussels data.*

Description

These data, introduced by Cook and Weisberg (1994), consist of 82 observations on horse mussels from New Zeland. The variables are shell length, width, height, mass and muscle mass

Usage

```
data(mussels)
```

Format

A data frame with 82 rows and 5 variables

myrng *Set seed for the MATLAB random number generator*

Description

Initializes the MATLAB random generator

Usage

```
myrng(seed)
```

Arguments

seed	a single value, interpreted as an integer

Value

Integer, the seed value with which the MATLAB random number generator was initialized.

Author(s)

FSDA team, <valentin.todorov@chello.at>

Examples

```
 ## Not run:
   data(wool)
   XX <- wool
   y <- XX[, ncol(XX)]
   X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

   seed <- myrng()                       #i nitialized the RNG and keep the seed

   myrng(seed)                           # repeat the computations with the same seed
   (out2 <- fsreg(X, y, method="LTS"))

   all.equal(out1, out2)


## End(Not run)
```

---

normBoxCox *Computes (normalized) Box-Cox transformation*

Description

Computes (normalized) Box-Cox transformation

Usage

```
normBoxCox(
  X,
  Col2Tra,
  la,
  Jacobian = TRUE,
  inverse = FALSE,
  bsb,
  trace = FALSE,
  ...
)
```

Arguments

X	The data matrix: n observations and p variables. The rows of X represent observations, and the columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
Col2Tra	Which variables to transform. An integer vector of length k specifying the variables which must be transformed. If it is missing and length(la)=p all variables are transformed

la	Transformation parameters. A vector of lenghth k containing set of transformation parameters for the k Col2Tra variables.
Jacobian	Requested Jacobian of transformed values. If Jacobian=TRUE the default, the transformation is normalized to have Jacobian equal to 1. Note that this optional argument is ignored if inverse=TRUE.
inverse	Wheather to return the inverse transformation. The default is inverse=FALSE.
bsb	Units to be used in the computation of the Jacobian, a vector of length m or or a logical vector of length n. The default value is bsb=1:n, i.e. all units are used to compute the Jacobian. Note that this option takes effect only if Jacobian=TRUE.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Details

When $\lambda \neq 0$ if Jacobian=TRUE,

$$ytra = (y^{\lambda} - 1)/(G^{(\lambda-1)}\lambda)$$

else if Jacobian=FALSE

$$ytra = (y^{\lambda} - 1)/\lambda$$

where G is the geometric mean of the observations.

When $\lambda = 0$ if Jacobian=TRUE,

$$ytra = Glog(y)$$

else if Jacobian=FALSE

$$ytra = log(y)$$

where G is the geometric mean of the observations.

Value

The transformed data matrix.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Box, G.E.P. and Cox, D.R. (1964), An analysis of transformations (with Discussion), "Journal of the Royal Statistical Society Series B", Vol. 26, pp. 211-252.

Examples

```
## Not run:
##  Mussels data.
data(mussels)

X <- mussels
la <- c(0.5, 0, 0.5, 0, 0)

## Transform all columns of the matrix X according to the values of la
(Y <- normBoxCox(X, la=la))

##  Check the inverse tranformation.
data(mussels)
X <- mussels
la <- c(0.5, 0, 0.5, 0, 0)

## Transform all columns of matrix Y according to the values of la
Ytra <- normBoxCox(X, la=la, Jacobian=FALSE)
Ychk <- normBoxCox(Ytra, la=la, inverse=TRUE)

print(max(max(abs(X-Ychk))))

## Comparison between Box-Cox and Yeo-Johnson transformation.
y <- seq(from=-2, to=2, by=0.1)
n <- length(y)
la <- seq(from=-1, to=3, by=1)
nla <- length(la)
YtraYJ <- matrix(0, nrow=n, ncol=nla)
YtraBC <- matrix(NA, nrow=n, ncol=nla)

posy <- y>0
for(j in 1:nla) {
  YtraYJ[,j] <- normYJ(y, 1, la[j], Jacobian=FALSE)
  YtraBC[posy, j] <- normBoxCox(y[posy], 1, la[j],Jacobian=FALSE)
}

oldpar <- par(mfrow=c(1,2))
plot(y, YtraYJ[,1], type="n", xlab="Original values",
    ylab="Transformed values", main="Yeo-Johnson transformation")
for(j in 1:nla)
    lines(y, YtraYJ[,j], col=j)
for(j in 1:nla) {
    text(y[1], YtraYJ[1,j], paste0("la=", la[j]))
}

plot(y, YtraBC[,1], type="n", xlab="Original values",
    ylab="Transformed values", main="Box-Cox transformation")
for(j in 1:nla)
    lines(y, YtraBC[,j], col=j)
for(j in 1:nla) {
    text(y[16], YtraBC[22,j], paste0("la=", la[j]))
}
```

```
 par(oldpar)

## End(Not run)
```

---

normYJ *Computes (normalized) Yeo-Johnson transformation*

Description

Computes (normalized) Yeo-Johnson transformation

Usage

```
normYJ(
  X,
  Col2Tra,
  la,
  Jacobian = TRUE,
  inverse = FALSE,
  bsb,
  trace = FALSE,
  ...
)
```

Arguments

X	The data matrix: n observations and p variables. The rows of X represent observations, and the columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
Col2Tra	Which variables to transform. An integer vector of length k specifying the variables which must be transformed. If it is missing and length(la)=p all variables are transformed
la	Transformation parameters. A vector of lenghth k containing set of transformation parameters for the k Col2Tra variables.
Jacobian	Requested Jacobian of transformed values. If Jacobian=TRUE the default, the transformation is normalized to have Jacobian equal to 1. Note that this optional argument is ignored if inverse=TRUE.
inverse	Wheather to return the inverse transformation. The default is inverse=FALSE.
bsb	Units to be used in the computation of the Jacobian, a vector of length m or or a logical vector of length n. The default value is bsb=1:n, i.e. all units are used to compute the Jacobian. Note that this option takes effect only if Jacobian=TRUE.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Details

The Yeo-Johnson transformation is the Box-Cox transformation of $y + 1$ for nonnegative values, and of $|y| + 1$ with parameter $2 - \lambda$ for y negative

Value

The transformed data matrix.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Yeo, I.K and Johnson, R. (2000), A new family of power transformations to improve normality or symmetry, "Biometrika", Vol. 87, pp. 954-959.

Examples

```
## Not run:

##  Example of use of normYJ() with all default options.
##  Transform value -3, -2, ..., 3
y <- (-3):3
lambda <- 0
y1 <- normYJ(y, Col2Tra=1, la=lambda)
plot(y, y1, xlab='Original values', ylab='Transformed values')


## Comparison between Box-Cox and Yeo-Johnson transformation.
y <- seq(from=-2, to=2, by=0.1)
n <- length(y)
la <- seq(from=-1, to=3, by=1)
nla <- length(la)
YtraYJ <- matrix(0, nrow=n, ncol=nla)
YtraBC <- matrix(NA, nrow=n, ncol=nla)

posy <- y>0
for(j in 1:nla) {
  YtraYJ[,j] <- normYJ(y, 1, la[j], Jacobian=FALSE)
  YtraBC[posy, j] <- normBoxCox(y[posy], 1, la[j],Jacobian=FALSE)
}

oldpar <- par(mfrow=c(1,2))
plot(y, YtraYJ[,1], type="n", xlab="Original values",
    ylab="Transformed values", main="Yeo-Johnson transformation")
for(j in 1:nla)
    lines(y, YtraYJ[,j], col=j)
for(j in 1:nla) {
    text(y[1], YtraYJ[1,j], paste0("la=", la[j]))
}
```

```
plot(y, YtraBC[,1], type="n", xlab="Original values",
    ylab="Transformed values", main="Box-Cox transformation")
for(j in 1:nla)
    lines(y, YtraBC[,j], col=j)
for(j in 1:nla) {
    text(y[16], YtraBC[22,j], paste0("la=", la[j]))
}

par(oldpar)


## End(Not run)
```

---

normYJpn *Computes (normalized) extended Yeo-Johnson transformation*

Description

Computes (normalized) extended Yeo-Johnson transformation

Usage

```
normYJpn(X, Col2Tra, la, Jacobian = TRUE, inverse = FALSE, trace = FALSE, ...)
```

Arguments

X	The data matrix: n observations and p variables. The rows of X represent observations, and the columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
Col2Tra	Which variables to transform. An integer vector of length k specifying the variables which must be transformed. If it is missing and length(la)=p all variables are transformed
la	Transformation parameters. A k x 2 matrix containing set of transformation parameters for the k Col2Tra variables. The first column contains the transformation parameter for positive observations and the second column the transformation parameter for negative observations.
Jacobian	Requested Jacobian of transformed values. If Jacobian=TRUE the default, the transformation is normalized to have Jacobian equal to 1. Note that this optional argument is ignored if inverse=TRUE.
inverse	Wheather to return the inverse transformation. The default is inverse=FALSE.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Details

The transformations for negative and positive responses were determined by Yeo and Johnson (2000) by imposing the smoothness condition that the second derivative of $zYJ(\lambda)$ with respect to y be smooth at $y = 0$. However some authors, for example Weisberg (2005), query the physical interpretability of this constraint which is oftern violated in data analysis. Accordingly, Atkinson et al (2019) and (2020) extend the Yeo-Johnson transformation to allow two values of the transformations parameter: λN for negative observations and λP for non-negative ones.

The Yeo-Johnson transformation is the Box-Cox transformation of $y + 1$ for nonnegative values, and of $|y| + 1$ with parameter $2 - \lambda$ for y negative

Value

The transformed data matrix.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C. Riani, M., Corbellini A. (2019), The analysis of transformations for profit-and-loss data, Journal of the Royal Statistical Society, Series C, "Applied Statistics", https://doi.org/10.1111/rssc.12389

Atkinson, A.C. Riani, M. and Corbellini A. (2021), The Box–Cox Transformation: Review and Extensions, "Statistical Science", Vol. 36, pp. 239-255, https://doi.org/10.1214/20-STS778

Yeo, I.K and Johnson, R. (2000), A new family of power transformations to improve normality or symmetry, "Biometrika", Vol. 87, pp. 954-959.

Examples

```
## Not run:

##  Example of use of normYJ() with all default options.
##  Transform value -3, -2, ..., 3
y <- (-3):3
lambda <- c(-0.5, 0.5)
y1 <- normYJpn(y, Col2Tra=1, la=lambda)
plot(y, y1, xlab='Original values', ylab='Transformed values')

## Compare Yeo and Johnson with extended Yeo and Yohnson.
## Transform value -3, -2, ..., 3
k <- 3
y <- seq(from=-k, to=k, by=0.01)

## Two values of lambda for extended Yeo and Johnson
lambda <- c(0, 0.5)
Jacobian <- FALSE
## Just one value of lambda for traditional Yao and Johnson
y1 <- normYJ(y,Col2Tra=1, la=lambda[1], Jacobian=Jacobian)
ypn <- normYJpn(y, Col2Tra=1, la=lambda,Jacobian=Jacobian)
```

```
plot(y, y1, type="l", xlab="Original values", ylab="Transformed values", col="blue")
lines(y, ypn, col="red")


## End(Not run)
```

---

poison *Poison*

Description

The poison data (by Box and Cox, 1964) are about the time to death of animals in a 3 x 4 factorial experiment with four observations at each factor combination. There are no outliers or influential observations that cannot be reconciled with the greater part of the data by a suitable transformation.

Usage

```
data(poison)
```

Format

A data frame with 48 rows and 7 variables: six explanatory and one response variable.

Source

G. E. P. Box and D. R. Cox (1964). An Analysis of Transformations, *Journal of the Royal Statistical Society. Series B*, **26**2 pp. 211–252.

Examples

```
data(poison)
head(poison)
```

---

psifun *Finds the tuning constant(s) associated to the supplied breakdown point or asymptotic efficiency for different psi functions*

Description

Finds the tuning constant(s) associated to the supplied breakdown point or asymptotic efficiency or computes breakdown point and efficiency associated with the supplied constant(s) for the following psi functions: TB=Tukey biweight, HA=Hampel, HU=Huber, HYP=Hyperbolic, OPT=Optimal, PD=mdpd.

Usage

```
psifun(
  u = vector(mode = "double", length = 0),
  p = 1,
  fun = c("TB", "bisquare", "biweight", "HA", "hampel", "HU", "huber", "HYP",
    "hyperbolic", "OPT", "optimal", "PD", "mdpd"),
  bdp,
  eff,
  const,
  param,
  trace = FALSE
)
```

Arguments

u	optional vector containing scaled residuals or Mahalanobis distances for the n units of the sample. If not provided, rho, psi, psider, psix and weights are not computed
p	number of variables (p=1 for regression)
fun	psi function class. One of TB, HA, HU, HP, OPT or PD.
bdp	requested breakdown point
eff	requested asymptotic efficiency
const	tuning constant c
param	additional parameters
trace	whether to print intermediate results. Default is trace=FALSE.

Value

A list will be returned containing the following elements:

1. class: link function which has be used. Possible values are 'bisquare', 'optimal', 'hyperbolic', 'hampel', 'huber' or 'mdpd'
2. bdp: breakdown point
3. eff: asymptotic efficiency
4. c1: consistency factor (and other parameters) associated to required breakdown point or nominal efficiency.
5. kc1: Expectation of rho associated with c1 to get a consistent estimator at the model distribution kc1 = E(rho) = sup(rho)*bdp
6. rho: vector of length n which contains the rho function associated to the residuals or Mahalanobis distances for the n units of the sample. Empty if u is not provided.
7. psi: vector of length n which contains the psi function associated with the residuals or Mahalanobis distances for the n units of the sample. Empty if u is not provided.
8. psider: vector of length n which contains the derivative of the psi function associated with the residuals or Mahalanobis distances for the n units of the sample. Empty if u is not provided.

9. psix: vector of length n which contains the psi function mutiplied by u associated with the residuals or Mahalanobis distances for the n units of the sample. Empty if u is not provided.
10. wei: vector of length n which contains the weights associated with the residuals or Mahalanobis distances for the n units of the sample. Empty if u is not provided.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Hoaglin, D.C. and Mosteller, F. and Tukey, J.W. (1982), *Understanding Robust and Exploratory Data Analysis*, Wiley, New York.

Huber, P.J. (1981), *Robust Statistics*, Wiley.

Huber, P.J. and Ronchetti, E.M. (2009), *Robust Statistics, 2nd Edition*, Wiley.

Hampel, F.R. and Rousseeuw, P.J. and Ronchetti E. (1981), The Change-of-Variance Curve and Optimal Redescending M-Estimators, *Journal of the American Statistical Association*, **76**, pp. 643–648.

Maronna, R.A. and Martin D. and Yohai V.J. (2006), *Robust Statistics, Theory and Methods*, Wiley, New York.

Riani, M. and Atkinson, A. C. and Corbellini, A. and Perrotta, D. (2020) Robust regression with density power divergence: Theory, comparisons, and data analysis, *Entropy* **22**. doi:10.3390/e22040399.

Examples

```
## Not run:
##  Find c for given bdp for the Tukey biweight function
##  The constant c associated to a breakdown point of
##  50 percent in regression is
##      c=1.547644980928226
    psifun(bdp=0.5)
    psifun(c=1.547644980928226)

##  Find c for given bdp for the Hampel function
psifun(bdp=0.5, fun="hampel")

## Plot Huber rho function.
x <- seq(-3, 3, 0.001)
c <- 1.345;

HUc1 <- psifun(u=x, p=1, fun="HU", const=c)
rhoHU <- HUc1$rho

plot(x, rhoHU, type="l", lty="solid", lwd=2, col="blue",
    xlab="u", ylab="rho (u,1.345)", ylim=c(0.16, 4.5))
lines(x, x^2/2, type="l", lty="dotted", lwd=1.5, col="red")
legend(-1, 4.6, legend=c("Huber rho function", "u^2/2"),
    lty=c("solid", "dotted"), lwd=c(2,1.5), col=c("blue", "red"))
```

```
yc <- 0.13;
text(-c, yc, paste0("-c=", -c), adj=1)
text(c,yc, paste0("c=",c), adj=0)

segments(c, 0, c, c**2/2, col="red")
segments(-c, 0, -c, c**2/2, col="red")
points(c, c**2/2, col="red")
points(-c, c**2/2, col="red")


## End(Not run)
```

---

regspmplot *Interactive scatterplot matrix for regression*

Description

Produces an interactive scatterplot of the responce `y` against each variable of the predictor matrix `X`.

Usage

```
regspmplot(
  y,
  X,
  group,
  plot,
  namey,
  nameX,
  col,
  cex,
  pch,
  labeladd,
  legend,
  xlim,
  ylim,
  tag,
  datatooltip,
  databrush,
  subsize,
  selstep,
  selunit,
  trace = FALSE,
  ...
)
```

Arguments

y	responce variable or an object containing the responce, the predictors and possibly other variable resulting from monitoring of regression. If y is a vector, a data matrix X must be present as an argument If y is a list containing just y and X, the call is equivallent to regspmplot(y, X). Otherwise y must be an an object of S3 class fsreda.object returned by fsreg with monitoring=TRUE - a list containing the monitoring along a search
X	Predictor variables. Data matrix of explanatory variables (also called 'regressors') of dimension n by p if the argument y is a vector. The rows of X represent observations, and the columns represent variables.
group	grouping variable. Vector with n elements. Specifies a grouping variable defined as a categorical variable (factor), numeric, or array of strings, or string matrix, and it must have the same number of rows as X. This grouping variable determines the marker and color assigned to each point. Remark: if group is used to distinguish a set of outliers from a set of good units, the id number for the outliers should be the larger (see optional field labeladd of parameter plo for details).
plot	This option controls the names which are displayed in the margins of the scatterplot matrix as well as the labels of the legend. If plot=FALSE, then namey, nameX and labeladd are both set to the empty string (default), and no label and no name is added to the plot. If plot=TRUE the names y, and X1,..., Xp are added to the margins of the the scatter plot matrix else nothing is shown. If plot is a list, it is possible to control not only the names but also, point labels, colors and symbols. More precisely list plot may contain the following elements: 1. labeladd - see parameter labeladd 2. namey - a character string containing the response variable name. See parameter namey. 3. nameX - a vector of character strings containing the labels of the explanatory variables. As default value, the labels which are added are Y1, ..., Yp. See parameter nameX. 4. clr - see parameter col 5. sym - see parameter pch 6. siz - see parameter cex 7. doleg - see parameter legend 8. xlimx - see parameter xlim 9. ylimy - see parameter ylim
namey	a character string with the name of the responce variable
nameX	a vector of character strings with the names of the explanatory variables
col	color specification for the data point. Can be different for each group. By default, the order of the colors is *blue*, *red*, *black*, *magenta*, *green*, *cyan* and *yelow*.
cex	the size of the symbols used for plotting. By default cex=1 the symbol size depends on the number of plots and the size of the figure window. Values larger than 1 will increase the size and values smaller than 1 will decrease the size.

pch — specification of the symbols to use. For example, if there are three groups, and pch=c(1, 3, 4), the first group will be plotted with a circle, the second with a plus, and the third with a 'x' (see ?pch or ?points for a list of symbols. NOTE: not all symbols available in R can be mapped to the symbols in MATLAB.

labeladd — logical, controls wheather the elements belonging to the last group in the scatterplot matrix are labelled with their unit row index or their rowname. The rowname is taken from the parameter label or if it is missing, from the sequence 1:n. The default value is labeladd=FALSE, i.e. no label is added.

legend — logical, controls where a legend is shown or not.

xlim — x limits. A vector with two elements controlling minimum and maximum on the x axis. By defaul automatic scale is used.

ylim — y limits. A vector with two elements controlling minimum and maximum on the y axis. By defaul automatic scale is used.

tag — Plot handle. String which identifies the handle of the plot which is about to be created. The default is tag='pl_mmd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

datatooltip — If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function datacursormode() for more details or see the examples below). The default options are DisplayStyle="Window" and SnapToDataVertex="on".

databrush — Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is TRUE or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. In addition, brushed units can be highlighted in the monitoring MD plot. Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too.

If databrush=TRUE the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If databrush=TRUE the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If databrush=list(...), it is possible to use all optional arguments of the MATLAB function selectdataFS() and the following optional arguments:

- persist: This option can be an empty value or a character containing 'on' or 'off'. The default value is persist="", that is brushing is allowed only once. If persist="on" or persis="off" brushing can be done as many time as the user requires. If persist='on' then the unit(s) currently brushed are added to those previously brushed. It is possible, every

time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.

- `labeladd`: add labels of brushed units in the scatterplot matrix. If this option is '1', we label the units of the last selected group with the unit row index in the matrix X. The default value is labeladd='', i.e. no label is added.

`subsize` x axis control, a numeric vector containing the subset size with length equal to the number of columns of matrix residuals. If it is not specified it will be set equal to `(nrow(residuals) - ncol(residuals) + 1) : nrow(residuals)`.

`selstep` Text shown in selected steps, a numeric vector which specifies for which steps of the forward search textlabels are added in the monitoring residual plot after a brushing action in the yXplot. The default is to write the labels at the initial and final step. The default is `selstep=c(m0, n)` where `m0` and `n` are respectively the first and final step of the search.

`selunit` Unit labelling. A vector of strings, a string, or a numeric vector for labelling units. If out is an object the threshold is associated with the trajectories of the residuals monitored along the search else it refers to the values of the response variable. If it is a vector of strings, only the lines associated with the units that in at least one step of the search had a residual smaller than `selunit[1]` or greater than `sellunit[2]` will have a textbox. If it is a string it specifies the threshold above which labels have to be put. For example `selunit='2.6'` means that the text labels are written only for the units which have in at least one step of the search a value of the scaled residual greater than 2.6 in absolute value. If it is a numeric vector it contains the list of the units for which it is necessary to put the text labels. The default value of selunit is string `'2.5'` if `y` is an object else it is an empty value.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

See Also

spmplot, mdrplot, resfwdplot

Examples

```
## Not run:
##  Example of the use of function regspmplot with all the default options
##  regsmplot() with first argument vector y and no option.
##  In the first example as input there are two matrices: y and X respectively
##  A simple plot is created
```

```
n <- 100
p <- 3
X <- matrix(data=rnorm(n*p), nrow=n, ncol=p)
y <- matrix(data=rnorm(n*1), nrow=n, ncol=1)
regspmplot(y, X)

##  Example of the use of function regspmplot with first argument
##  vector y and third argument group.
##  Different groups are shown in the yXplot

group <- rep(0, n)
group[1:(n/2)] <- rep(1, n/2)
regspmplot(y, X, group)

##  Example of the use of function regspmplot with first argument
##  vector y, third argument group and fourth argument plot
##  (Ex1) plot=TRUE

regspmplot(y, X, group, plot=TRUE)

##  (Ex1) Set the scale for the x axes, the y axis and control symbol type
regspmplot(y, X, group, xlim=c(-1,2), ylim=c(0,2), pch=c(10,11), trace=TRUE)

##  When the first input argument is an object.
##  In the following example the input is an object which also contains
##  information about the forward search.
    (out <- fsreg(y~X, method="LMS", control=LXS_control(nsamp=1000)))
    (out <- fsreg(y~X, bsb=out$bs, monitoring=TRUE))

    regspmplot(out, plot=0)


## End(Not run)
```

`resfwdplot` *Plots the trajectories of the monitored scaled (squared) residuals*

Description

Plots the trajectories of the monitored scaled (squared) residuals

Usage

```
resfwdplot(out,
    xlim, ylim, xlab, ylab, main, lwd, lty, col, cex.lab, cex.axis,
    xvalues,
    fg.thresh, fg.unit, fg.labstep, fg.lwd, fg.lty, fg.col, fg.mark, fg.cex,
    bg.thresh, bg.style,
```

```
tag, datatooltip, label, nameX, namey, msg, databrush,
standard, fground, bground, ...)
```

Arguments

out	An object returned by one of the monitoring functions (see FSReda_control, Sregeda_control and MMregeda_control). The object is one of fsreda.object, sregeda.object or mmregeda.object. The needed elements of out are 1. RES: matrix containing the residuals monitored in each step of the forward search or any other robust procedure. Every row is associated with a residual (unit). This matrix can be created using function FSReda, Sregeda, MMregeda. 2. Un: (for FSR only) - matrix containing the order of entry in the subset of each unit (required only when datatooltip is true or databrush is not empty). 3. bdp: (for Sreg only) - vector containing a sequence of breakdown point values to monitor on. 4. eff: (for MMreg only) - vector containing a sequence of efficiency values to monitor on. 5. y: a vector containing the response (required only when option databrush is requested). 6. X: a matrix containing the explanatory variables (required only when option databrush is requested). 7. Bols: (n-init+1) x (p+1) matrix containing the estimated beta coefficients monitored in each step of the robust procedure (required only when option databrush is requested and suboption multivarfit is requested).
ylim	Control y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
xlim	Control x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
lwd	The line width, a positive number, defaulting to 1
lty	The line type. Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.
col	colors to be used for the highlighted units
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
xvalues	values for the x axis. Numeric vector of ncol(RES) controlling the x axis coordinates. The default value of xvalues is (nrow(RES) - ncol(RES) + 1):nrow(RES)

`fg.thresh` (alternative to fg.unit) numeric vector of length 1 or 2 which specifies the highlighted trajectories. If `length(fthresh) == 1` the highlighted trajectories are those of units that throughtout the search had at leat once a residual greater (in absolute value) than thresh. The default value is `fg.thresh=2.5`. If `length(fthresh) == 2` the highlighted trajectories are those of units that throughtout the search had a residual at leat once bigger than `fg.thresh[2]` or smaller than `fg.thresh[1]`.

`fg.unit` (alternative to fg.thresh), vector containing the list of the units to be highlighted. If `fg.unit` is supplied, `fg.thresh` is ignored.

`fg.labstep` numeric vector which specifies the steps of the search where to put labels for the highlighted trajectories (units). The default is to put the labels at the initial and final steps of the search. `fg.labstep=''` means no label.

`fg.lwd` The line width for the highlighted trajectories (units). Default is 1.

`fg.lty` The line type for the highlighted trajectories (units). Line types can either be specified as an integer (1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash". The latter two are not supported by Matlab.

`fg.col` colors to be used for the highlighted units.

`fg.mark` Controlls whether to plot highlighted trajectories as symbols. if `fg.mark==TRUE` each line is plotted using a different symbol else no marker is used (default).

`fg.cex` controls the font size of the labels of the trajectories in foreground.

`bg.thresh` numeric vector of length 1 or 2 which specifies how to define the unimmportant trajectories. Unimmportant trajectories will be plotted using a colormap, in greysh or will be hidden. If `length(thresh) == 1` the irrelevant units are those which always had a residual smaller (in absolute value) than thresh. If `length(bthresh) == 2` the irrelevant units are those which always had a residual greater than bthresh(1) and smaller than bthresh(2). The default is: `bg.thresh=2.5` if `n > 100` and `bg.thresh=-Inf` if `n <= 100` i.e. to treat all trajectories as important if `n <= 100` and, if `n > 100`, to reduce emphasis only to trajectories having in all steps of the search a value of scaled residual smaller than 2.5.

`bg.style` specifies how to plot the unimportant trajectories as defined in option bthresh.

1. `bg.style="faint"`: unimportant trajectories are plotted using a colormap.
2. `bg.style="hide"`: unimportant trajectories are hidden.
3. `bg.style="greyish"`: unimportant trajectories are displayed in a faint grey.

When `n>100` the default option is `bg.style='faint'`. When `n <= 100` and `bg.thresh == -Inf` option bstyle is ignored. Remark: bground=" is equivalent to -Inf that is all trajectories are considered relevant.

`tag` Plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_resfwd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

`datatooltip` Interactive clicking. It is inactive if this parameter is missing or empty. The default is `datatooltip=TRUE`, i.e. the user can select with the mouse an individual residual trajectory in order to have information about the corresponding unit. The information displayed depends on the estimator in use.

For example for class `fsreda.object` the information concerns the label and the step of the search in which the unit enters the subset. If datatooltip is a list it may contain the following fields:

1. DisplayStyle determines how the data cursor displays. Possible values are `'datatip'` and `'window'` (default). `'datatip'` displays data cursor information in a small yellow text box attached to a black square marker at a data point you interactively select. `'window'` displays data cursor information for the data point you interactively select in a floating window within the figure.
2. SnapToDataVertex: specifies whether the data cursor snaps to the nearest data value or is located at the actual pointer position. Possible values are `SnapToDataVertex='on'` (default) and `SnapToDataVertex='off'`.
3. LineColor: controls the color of the trajectory selected with the mouse. It can be an RGB triplet of values between 0 and 1, or character vector indicating a color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation.
4. SubsetLinesColor: enables to control the color of the trajectories of the units that are in the subset at a given step of the search (if `resfwdplot()` is applied to an object of class `fsreda.object`) or have a weight greater than 0.9 (if `resfwdplot()` is applied to an object of class `sregeda.object` or `mmregeda.object`). This can be done (repeatedly) with a left mouse click in proximity of the step of interest. A right mouse click will terminate the selection by marking with a up-arrow the step corresponding to the highlighted lines. The highlighted lines by default are in red, but a different color can be specified as RGB triplet or character of color name. Note that a RGB vector can be conveniently chosen with our MATLAB class FSColor, see documentation. By default `SubsetLinesColor=""`, i.e. the modality is not active. Any initialization for `SubsetLinesColor` which cannot be interpreted as RGB vector will be converted to blue, i.e. `SubsetLinesColor` will be forced to be [0 0 1]. If `SubsetLinesColor` is not empty the previous option `LineColor` is ignored.

`label` Character vector containing the labels of the units (optional argument used when `datatooltip=TRUE`). If this field is not present labels row1, ..., rown will be automatically created and included in the pop up datatooltip window).

`nameX` Add variable labels in plot. A vector of strings of length p containing the labels of the variables of the regression dataset. If it is empty (default) the sequence `X1, ..., Xp` will be created automatically

`namey` Add response label. A string containing the label of the response

`msg` Controls whether to display or not messages on the screen If `msg==1` (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

`databrush` interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is a scalar or a list) enables

the user to select a set of trajectories in the current plot and to see them highlighted in the y|X plot, i.e. a matrix of scatter plots of y against each column of X, grouped according to the selection(s) done by brushing. If the plot y|X does not exist it is automatically created. In addition, brushed units are automatically highlighted in the minimum deletion residual plot if it is already open. The extension to the following plots will be available in future versions of the toolbox:

1. monitoring leverage plot;
2. maximum studentized residual;
3. s^2 and R^2;
4. Cook distance and modified Cook distance;
5. deletion t statistics.

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If `databrush=list(...)`, it is possible to use all optional arguments of function `selectdataFS()` and the following optional argument:

1. `persist`. Persist is an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
2. `bivarfit`. Wheather to superimpose bivariate least square lines on the plot (if `plot=TRUE`. This option adds one or more least squares lines, based on SIMPLE REGRESSION of y on Xi, to the plots of y|Xi. The default is `bivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. If `bivarfit=2`, two OLS lines are fitted: one to all points and another to the group of the genuine observations. The group of the potential outliers is not fitted. If `bivarfit=0` one OLS line is fitted to each group. This is useful for the purpose of fitting mixtures of regression lines. If `bivarfit='i1'` or `bivarfit='i2'`, etc. an OLS line is fitted to a specific group, the one with index 'i' equal to 1, 2, 3 etc. Again, useful in case of mixtures.
3. `multivarfit`. Wheather to superimpose multivariate least square lines. This option adds one or more least square lines, based on MULTIVARIATE REGRESSION of y on X, to the plots of y|Xi. The default is `multivarfit=FALSE`: no line is fitted. If `bivarfit=1`, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. The line added to the scatter plot y|Xi is avconst + Ci*Xi, where Ci is the coefficient of Xi in the multivariate regression and avconst is the effect of all the other explanatory variables different from Xi evaluated at their centroid (that is overline(y)'C)). If `multivarfit=2`, same action as with `multivarfit=1` but this time we also

add the line based on the group of unselected observations (i.e. the normal units).

4. `labeladd`. Add outlier labels in plot. If `labeladd=TRUE`, we label the outliers with the unit row index in matrices X and y. The default value is `labeladd=FALSE`, i.e. no label is added.

`standard`	(MATLAB-style arguments) appearance of the plot in terms of xlim, ylim, axes labels and their font size style, color of the lines, etc.
`fground`	MATLAB-style arguments for the fground trajectories in foregroud.
`bground`	MATLAB-style arguments for the fground trajectories in backgroud.
`...`	potential further arguments passed to lower level functions.

Details

No details

Value

No value returned

Author(s)

FSDA team

Examples

```
## Not run:

n <- 100
y <- rnorm(n)
X <- matrix(rnorm(n*4), nrow=n)

out <- fsreg(y~X, method="LTS")
out <- fsreg(y~X, method="FS", bsb=out$bs, monitoring=TRUE)
resfwdplot(out)


## End(Not run)
```

`resindexplot` *Plots the residuals from a regression analysis versus index number or any other variable*

Description

The function `resindexplot()` plots the residuals from a regression analysis versus index number or any other variable. The residuals come from an output object of any of the regression fucntions or a simply a vector of values. In order to use the databrush option, the residuals must come from one of the fsdaR regression functions.

Usage

```
resindexplot(out, x, xlim, ylim, xlab, ylab, main, numlab, indlab, conflev, cex.axis,
    cex.lab, lwd, nameX, namey, tag, col, cex, databrush, ...)
```

Arguments

out	A vector containing the residuals from a regression analysis or an object returned by one of the regression functions (see FSR_control, LXS_control, Sreg_control and MMreg_control). The object is one of fsr.object, fsdalts.object, fsdalms.object, sreg.object or mmreg.object. The needed elements of out are at least residuals, but if the option databrush is used, also X amd y will be needed.
x	The vector to be plotted on the x-axis. As default the sequence 1:length(residuals) will be used
xlim	Control x scale in plot. Vector with two elements controlling minimum and maximum on the x axis. Default is to use automatic scale.
ylim	Control y scale in plot. Vector with two elements controlling minimum and maximum on the y axis. Default is to use automatic scale.
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
numlab	Number of points to be identified in plots (see also indlab) . By default the five points with largest values will be identified. If numlab is a single number containing scalar k, the units with the k largest residuals are labelled in the plots. If numlab is a vector, the units inside vector numlab are labelled in the plots. The default value of numlab=5 and the units with the 5 largest residuals will be labelled. If numlib=0 or numlib=NULL no labelling will be done.
indlab	Which points to be identified in plots (see also numlab) - the units with indexes in the vector indlab are labelled in the plots.
conflev	Confidence interval for the horizontal bands (a numeric vector). It can be a vector of different confidence level values. Remark: confidence interval is based on the chi^2 distribution
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex
cex.lab	The magnification to be used for x and y labels relative to the current setting of cex
lwd	The line width, a positive number, defaulting to 1
tag	Figure tag (character). Tag of the figure which will host the resindexplot. The default tag is pl_resindex.
col	Fill color for markers that are closed shapes (circle, square, diamond, pentagram, hexagram, and the four triangles). Can be 'none' or 'auto' or color name(string) or RGB triplet.
cex	Size of the point symbols. The magnification to be used relative to the current setting of cex.

nameX — Add variable labels in plot. A vector of strings of length p containing the labels of the variables of the regression dataset. If it is empty (default) the sequence X1, ..., Xp will be created automatically

namey — Add response label. A string containing the label of the response

databrush — Interactive mouse brushing. If databrush is missing or empty (default) or databrush=FALSE, no brushing is done. The activation of this option (databrush is a scalar or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the y|X plot, i.e. a matrix of scatter plots of y against each column of X, grouped according to the selection(s) done by brushing. If the plot y|X does not exist it is automatically created. In addition, brushed units are automatically highlighted in the minimum deletion residual plot if it is already open. The extension to the following plots will be available in future versions of the package:

1. monitoring leverage plot;
2. maximum studentized residual;
3. s^2 and R^2;
4. Cook distance and modified Cook distance;
5. deletion t statistics.

Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too

If databrush=TRUE the default selection tool is a rectangular brush and it is possible to brush only once (that is persist='').

If databrush=list(...), it is possible to use all optional arguments of function selectdataFS() and the following optional argument:

1. persist. Persist is an empty value or a character containing 'on' or 'off'. The default value is persist="", that is brushing is allowed only once. If persist="on" or persis="off" brushing can be done as many time as the user requires. If persist='on' then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If persist='off' every time a new brush is performed units previously brushed are removed.
2. bivarfit. This option adds one or more least square lines based on SIMPLE REGRESSION to the plots of y|X, depending on the selected groups. The default is bivarfit=FALSE: no line is fitted. If bivarfit=1, a single OLS line is fitted to all points of each bivariate plot in the scatter matrix y|X. If bivarfit=2, two OLS lines are fitted: one to all points and another to the group of the genuine observations. The group of the potential outliers is not fitted. If bivarfit=0 one OLS line is fitted to each group. This is useful for the purpose of fitting mixtures of regression lines. If bivarfit='i1' or bivarfit='i2', etc. an OLS line is fitted to a specific group, the one with index 'i' equal to 1, 2, 3 etc. Again, useful in case of mixtures.
3. multivarfit. Wheather to superimpose multivariate least square lines. This option adds one or more least square lines, based on MULTIVARIATE REGRESSION of y on X, to the plots of y|Xi. The default is multivarfit=FALSE: no line is fitted. If bivarfit=1, a single OLS line is fitted to all points of

each bivariate plot in the scatter matrix y|X. The line added to the scatter plot y|Xi is avconst + Ci*Xi, where Ci is the coefficient of Xi in the multivariate regression and avconst is the effect of all the other explanatory variables different from Xi evaluated at their centroid (that is overline(y)'C)). If `multivarfit=2`, same action as with `multivarfit=1` but this time we also add the line based on the group of unselected observations (i.e. the normal units).

4. `labeladd`. Add outlier labels in plot. If `labeladd=TRUE`, we label the outliers with the unit row index in matrices X and y. The default value is `labeladd=FALSE`, i.e. no label is added.

... potential further arguments passed to lower level functions.

Details

No details

Value

No value returned

Author(s)

FSDA team

Examples

```
## Not run:
out <- fsreg(stack.loss~., data=stackloss)
resindexplot(out, conflev=c(0.95,0.99), col="green")

## End(Not run)
```

---

score *Computes the score test for transformation in regression*

Description

Computes the score test for transformation in regression

Usage

```
score(x, ...)

## S3 method for class 'formula'
score(
  formula,
  data,
  subset,
```

```
  weights,
  na.action,
  model = TRUE,
  contrasts = NULL,
  offset,
  ...
)

## Default S3 method:
score(
  x,
  y,
  intercept = TRUE,
  la = c(-1, -0.5, 0, 0.5, 1),
  lik = FALSE,
  nocheck = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
...	potential further arguments passed to lower level functions.
formula	a formula of the form y ~ x1 + x2 + ...
data	data frame from which variables specified in formula are to be taken.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used **NOT USED YET**.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The "factory-fresh" default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
model	logical indicating if the model frame, is to be returned.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. An offset term can be included in the
y	Response variable. A vector with n elements that contains the response variable.
intercept	wheather to use constant term (default is intercept=TRUE
la	values of the transformation parameter for which it is necessary to compute the score test. Default value of lambda is la=c(-1, -0.5, 0, 0.5, 1), i.e., the five most common values of lambda.

lik	likelihood for the augmented model. If true the value of the likelihood for the augmented model will be calculated and returend otherwise (default) only the value of the score test will be given
nocheck	Whether to check input arguments. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. The default is nocheck=FALSE.
trace	Whether to print intermediate results. Default is trace=FALSE.

Value

An S3 object of class score.object will be returned which is basically a list containing the following elements:

1. la: transformation parameter - vector containing the values of lambda for which the score test is performed.
2. Score: a vector containing the values of the score test for each value of the transformation parameter.
3. Lik: value of the likelihood. This output is produced only if lik=TRUE.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Atkinson, A.C. and Riani, M. (2000), *Robust Diagnostic Regression Analysis* Springer Verlag, New York.

Examples

```
## Not run:
  data(wool)
  XX <- wool
  y <- XX[, ncol(XX)]
  X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

  (out <- score(X, y))                        # call 'score' with all default parameters
  (out <- score(cycles~., data=wool))           # use the formula interface
  (out <- score(cycles~., data=wool, lik=TRUE)) # return the likelihood

  data(loyalty)
  head(loyalty)
  ##    la is a vector containing the values of \lambda which have to be tested
  (out <- score(amount_spent~., data=loyalty, la=c(0.25, 1/3, 0.4, 0.5)))
  (out <- score(amount_spent~., data=loyalty, la=c(0.25, 1/3, 0.4, 0.5), lik=TRUE))

## End(Not run)
```

score.object *Objects returned by the function* score

Description

An object of class score.object holds information about the result of a call to score.

Value

The functions print() and summary() are used to obtain and print a summary of the results. An object of class score is a list containing at least the following components:

1. la: vector containing the values of lambda for which the score test is computed
2. Score: a vector containing the values of the score test for each value of the transformation parameter.
3. Lik: value of the likelihood. This output is produced only if lik=TRUE.

Examples

```
 ## Not run:
   data(wool)
   (out <- score(cycles~., data=wool, lik=TRUE))

   class(out)
   summary(out)

## End(Not run)
```

scoreYJ *Computes the score test for Yeo and Johnson transformation*

Description

Computes the score test for Yeo and Johnson transformation

Usage

```
scoreYJ(
  x,
  y,
  intercept = TRUE,
  la = c(-1, -0.5, 0, 0.5, 1),
  lik = FALSE,
  nocheck = FALSE,
  tukey1df = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
y	Response variable. A vector with n elements that contains the response variable.
intercept	wheather to use constant term (default is intercept=TRUE
la	values of the transformation parameter for which it is necessary to compute the score test. Default value of lambda is la=c(-1, -0.5, 0, 0.5, 1), i.e., the five most common values of lambda.
lik	likelihood for the augmented model. If true the value of the likelihood for the augmented model will be calculated and returend otherwise (default) only the value of the score test will be given
nocheck	Whether to check input arguments. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. The default is nocheck=FALSE.
tukey1df	Tukey's one degree of freedome test for non-additivity. The constructed variable is given by $$w_T(\lambda) = (\hat{z}(\lambda) - \overline{z}(\lambda))^2 / 2\overline{z}(\lambda)$$ where $z(\lambda)$ is the transformed response, and $\hat{z}(\lambda)$ are the fitted values on the transformed response. The t test on the constructed variable above provides a test from departures from the assumed linear model and is known in the literature as Tukey's one degree of freedome test for non-additivity. If tukey1df=TRUE the test is computed and returned in ScoreT else (default) the value of the test is not computed.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

An S3 object of class scoreYJ.object will be returned which is basically a list containing the following elements:

- Score: score test. A vector of length length(lambda) which contains the value of the score test for each value of lambda specified in the optional input parameter la. If la is not specified, the vector will be of length 5 and contains the values of the score test for the 5 most common values of lambda.
- ScoreT: value of the Tukey's one degree of freedome test for non-additivity. This output is produced only if tukey1df=TRUE.
- Lik: value of the likelihood. This output is produced only if lik=TRUE.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Yeo, I.K. and Johnson, R. (2000), A new family of power transformations to improve normality or symmetry, "Biometrika", Vol. 87, pp. 954-959.

Examples

```
## Not run:
##  ScoreYJ with all default options for the wool data.
##  Load the wool data.

  data(wool)
  XX <- wool
  y <- XX[, ncol(XX)]
  X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

  (out <- scoreYJ(X, y))                    # call 'scoreYJ' with all default parameters
  (out <- scoreYJ(X, y, lik=TRUE))          # return the likelihood

  data(loyalty)
  head(loyalty)
  y=loyalty[, 4]
  X=loyalty[, 1:3]

  ##    la is a vector containing the values of \lambda which have to be tested
  (out <- scoreYJ(X, y, la=c(0.25, 1/3, 0.4, 0.5)))
  (out <- scoreYJ(X, y, la=c(0.25, 1/3, 0.4, 0.5), lik=TRUE))

## End(Not run)
```

`scoreYJ.object` *Objects returned by the function* `scoreYJ`

Description

An object of class `scoreYJ.object` holds information about the result of a call to `scoreYJ`.

Value

The functions `print()` and `summary()` are used to obtain and print a summary of the results. An object of class `scoreYJ` is a list containing at least the following components:

- `la`: vector containing the values of lambda for which fan plot is constructed
- `Score`: a vector containing the values of the score test for each value of the transformation parameter.
- `ScoreT`: value of the Tukey's one degree of freedome test for non-additivity. This output is produced only if `tukey1df=TRUE`.
- `Lik`: value of the likelihood. This output is produced only if `lik=TRUE`.

Examples

```
## Not run:
  data(wool)
 XX <- wool
 y <- XX[, ncol(XX)]
 X <- XX[, 1:(ncol(XX)-1), drop=FALSE]

 (out <- scoreYJ(X, y, lik=TRUE))
 class(out)
 summary(out)

## End(Not run)
```

`scoreYJpn` *Computes the score test for YJ transformation for pos and neg observations*

Description

The transformations for negative and positive responses were determined by Yeo and Johnson (2000) by imposing the smoothness condition that the second derivative of `zYJ(lambda)` with respect to `y` be smooth at `y = 0`. However, some authors, for example Weisberg (2005), query the physical interpretability of this constraint which is often violated in data analysis. Accordingly, Atkinson et al (2019) and (2020) extend the Yeo-Johnson transformation to allow two values of the transformations parameter: `lambda N` for negative observations and `lambda P` for non-negative ones. ScoreYJpn computes:

- 1) the t test associated with the constructed variable computed assuming a different transformation for positive observations keeping the value of the transformation parameter for negative observations fixed. In short, we call this test 'test for positive observations'.
- 2) the t test associated with the constructed variable computed assuming a different transformation for negative observations keeping the value of the transformation parameter for positive observations fixed. In short, we call this test 'test for negative observations'.
- 3) the F test for the joint presence of the two constructed variables described in points 1) and 2).

Usage

```
scoreYJpn(
  x,
  y,
  intercept = TRUE,
  la = c(-1, -0.5, 0, 0.5, 1),
  nocheck = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
y	Response variable. A vector with n elements that contains the response variable.
intercept	wheather to use constant term (default is intercept=TRUE
la	values of the transformation parameter for which it is necessary to compute the score test. Default value of lambda is la=c(-1, -0.5, 0, 0.5, 1), i.e., the five most common values of lambda.
nocheck	Whether to check input arguments. If nocheck=TRUE no check is performed on matrix y and matrix X. Notice that y and X are left unchanged. In other words the additional column of ones for the intercept is not added. The default is nocheck=FALSE.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

An S3 object of class scoreYJpn.object will be returned which is basically a list containing the following elements:

- la: vector containing the values of lambda for which the test is computed
- Score: Matrix of size length(lambda)-by-3 that contains the value of the score test for each value of lambda specified in optional input parameter la. The first column refers to the test for positive observations, the second column refers to the test for negative observations and the third column refers to the F test for the joint presence of the two constructed variables. If la is not specified, the number of rows of the matrix Score is equal to 5 and will contain the values of the score test for the 5 most common values of lambda.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Yeo, I.K. and Johnson, R. (2000), A new family of power transformations to improve normality or symmetry, "Biometrika", Vol. 87, pp. 954-959.

Atkinson, A.C. Riani, M., Corbellini A. (2019), The analysis of transformations for profit-and-loss data, Journal of the Royal Statistical Society, Series C, "Applied Statistics", https://doi.org/10.1111/rssc.12389

Atkinson, A.C. Riani, M. and Corbellini A. (2021), The Box–Cox Transformation: Review and Extensions, "Statistical Science", Vol. 36, pp. 239-255, https://doi.org/10.1214/20-STS778

Examples

```
## Not run:
##  Example in which positive and negative observations require the same lambda
    set.seed(1234)
    n <- 100
    y <- rnorm(n)
    ## Transform the value to find out if we can recover the true value of
    ## the transformation parameter.
    la <- 0.5
    ytra <- normYJ(y, la=la, inverse=TRUE)

    ## Start the analysis
    X <- rep(1, n)
    outSC <- scoreYJ(X, ytra, intercept=FALSE)
    outSCpn <- scoreYJpn(X, ytra, intercept=FALSE)
    la <- c(-1, -0.5, 0, 0.5, 1)
    print(cbind(la,  outSCpn$Score[,1], outSC$Score, outSCpn$Score[,2]))

    ##  Comment: if we consider the 5 most common values of lambda,
    ##  the value of the score test when lambda=0.5 is the only one which is not
    ##  significant. Both values of the score test for positive and negative
    ##  observations confirm that this value of the transformation parameter is
    ##   OK for both sides of the distribution.


## End(Not run)
```

`scoreYJpn.object` *Objects returned by the function* `scoreYJpn`

Description

An object of class `scoreYJpn.object` holds information about the result of a call to `scoreYJpn`.

Value

The functions `print()` and `summary()` are used to obtain and print a summary of the results. An object of class `scoreYJpn` is a list containing at least the following components:

- `la`: vector containing the values of lambda for which the test is computed
- `Score`: Matrix of size `length(lambda)-by-3` that contains the value of the score test for each value of lambda specified in optional input parameter `la`. The first column refers to the test for positive observations, the second column refers to the test for negative observations and the third column refers to the F test for the joint presence of the two constructed variables. If `la` is not specified, the number of rows of the matrix `Score` is equal to 5 and will contain the values of the score test for the 5 most common values of `lambda`.

Examples

```
 ## Not run:
     set.seed(1234)
     n <- 100
     y <- rnorm(n)
     ## Transform the value to find out if we can recover the true value of
     ## the transformation parameter.
     la <- 0.5
 ##    ytra <- normYJ(y,[],la,'inverse',true);
     ytra <- y

     ## Start the analysis
     X <- rep(1, n)
     (outSCpn <- scoreYJpn(X, ytra, intercept=FALSE))
     class(outSCpn)
     summary(outSCpn)

## End(Not run)
```

`smult` *Computes S estimators in multivariate analysis*

Description

Computes S estimators in multivariate analysis

Usage

```
smult(
  x,
  monitoring = FALSE,
  plot = FALSE,
  bdp,
  nsamp,
  conflev = 0.975,
  nocheck = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.

`monitoring` Wheather to perform monitoring of Mahalanobis distances and other specific quantities

`plot` Plots the Mahalanobis distances against index number. If `plot=FALSE` (default) or `plot=0` no plot is produced. The confidence level used to draw the confidence bands for the MD is given by the input option conflev. If conflev is not specified a nominal 0.975 confidence interval will be used. If `plot=2` a scatter plot matrix with the outliers highlighted is produced. If plot is a list it may contain the following fields:

- labeladd If `labeladd=1`, the outliers in the spm are labelled with the unit row index. The default value is `labeladd=""`, i.e. no label is added
- nameY character vector containing the labels of the variables. As default value, the labels which are added are Y1, ...Yp.

`bdp` Measures the fraction of outliers the algorithm should resist. In this case any value greater than 0 but smaller or equal than 0.5 will do fine (default is `bdp=0.5`). Note that given bdp nominal efficiency is automatically determined.

`nsamp` Number of subsamples which will be extracted to find the robust estimator. If `nsamp=0` all subsets will be extracted. They will be `(n choose p)`. If the number of all possible subset is `<1000` the default is to extract all subsets otherwise just 1000.

`conflev` Confidence level which is used to declare units as outliers (scalar). Usually `conflev=0.95`, `conflev=0.975` or `conflev=0.99` (individual alpha) `conflev=1-0.05/n`, `conflev=1-0.025/n` or `conflev=1-0.01/n` (simultaneous alpha). Default value is `convlev=0.975`.

`nocheck` It controls whether to perform checks on matrix Y. If `nocheck=TRUE`, no check is performed.

`trace` Whether to print intermediate results. Default is `trace=FALSE`.

`...` potential further arguments passed to lower level functions.

Details

This function follows the lines of MATLAB/R code developed during the years by many authors. For more details see http://www.econ.kuleuven.be/public/NDBAE06/programs/ and the R package CovSest The core of these routines, e.g. the resampling approach, however, has been completely redesigned, with considerable increase of the computational performance.

Value

Depending on the input parameter `monitoring`, one of the following objects will be returned:

1. `smult.object`
2. `smulteda.object`

Author(s)

FSDA team, `<valentin.todorov@chello.at>`

References

Maronna, R.A., Martin D. and Yohai V.J. (2006), Robust Statistics, Theory and Methods, Wiley, New York.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- smult(hbk[,1:3]))
class(out)
summary(out)

##  Generate contaminated data (200,3)
n <- 200
p <- 3
set.seed(123456)
X <- matrix(rnorm(n*p), nrow=n)
Xcont <- X
Xcont[1:5, ] <- Xcont[1:5,] + 3

out1 <- smult(Xcont, trace=TRUE)           # no plots (plot defaults to FALSE)
names(out1)

## plot=TRUE - generates: (1) a plot of Mahalanobis distances against
##    index number. The confidence level used to draw the confidence bands for
##    the MD is given by the input option conflev. If conflev is
##    not specified a nominal 0.975 confidence interval will be used and
##    (2) a scatter plot matrix with the outliers highlighted.

(out1 <- smult(Xcont, trace=TRUE, plot=TRUE))

## plots is a list: the spm shows the labels of the outliers.
(out1 <- smult(Xcont, trace=TRUE, plot=list(labeladd="1")))

## plots is a list: the spm uses the variable names provided by 'nameY'.
(out1 <- smult(Xcont, trace=TRUE, plot=list(nameY=c("A", "B", "C"))))

## smult() with monitoring
(out2 <- smult(Xcont, monitoring=TRUE, trace=TRUE))
names(out2)

##  Forgery Swiss banknotes examples.

data(swissbanknotes)

(out1 <- smult(swissbanknotes[101:200,], plot=TRUE))

(out1 <- smult(swissbanknotes[101:200,], plot=list(labeladd="1")))

## End(Not run)
```

smult.object *Description of* smult.object *Objects*

Description

An object of class smult.object holds information about the result of a call to smult.

Value

The object itself is basically a list with the following components:

loc	p-by-1 vector containing S estimate of location.
shape	p-by-p matrix containing robust estimate of the shape matrix. Remark: det|shape|=1.
scale	robust estimate of the scale.
cov	scale^2 * shape: robust estimate of covariance matrix.
bs	a (p+1) vector containing the units forming best subset associated with S estimate of location.
md	n-by-1 vector containing the estimates of the robust Mahalanobis distances (in squared units). This vector contains the distances of each observation from the location of the data, relative to the scatter matrix cov.
outliers	A vector containing the list of the units declared as outliers using confidence level specified in input scalar conflev.
conflev	Confidence level that was used to declare outliers.
singsub	Number of subsets without full rank. Notice that singsub > 0.1*(number of subsamples) produces a warning.
weights	n x 1 vector containing the estimates of the weights.
X	the data matrix X

The object has class "smult".

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- smult(hbk[,1:3]))
class(out)
summary(out)

## End(Not run)
```

smulteda.object *Description of* smulteda.object *Objects*

Description

An object of class smulteda.object holds information about the result of a call to smult with monitoring=TRUE.

Value

The object itself is basically a list with the following components:

Loc	length(bdp)-by-p matrix containing S estimate of location for each value of bdp.
Shape	p-by-p-by-length(bdp) 3D array containing robust estimate of the shape for each value of bdp. Remark: det|shape|=1.
Scale	length(bdp) vector containing robust estimate of the scale for each value of bdp.
Cov	p-by-p-by-length(bdp) 3D array containing robust estimate of covariance matrix for each value of bdp. Note that scale(i)^2 * shape[,,i] = robust estimate of covariance matrix.
Bs	(p+1)-by-length(bdp) matrix containing the units forming best subset for each value of bdp.
MAL	n-by-length(bdp) matrix containing the estimates of the robust Mahalanobis distances (in squared units) for each value of bdp.
Outliers	n-by-length(bdp) matrix containing flags for the outliers. Boolean matrix containing the list of the units declared as outliers for each value of bdp using confidence level specified in input scalar conflev
Weights	n x length(bdp) matrix containing the weights for each value of bdp.
conflev	Confidence level that was used to declare outliers.
singsub	Number of subsets without full rank. Notice that singsub > 0.1*(number of subsamples) produces a warning.
bdp	vector which contains the values of bdp which have been used.
X	the data matrix X.

The object has class "smulteda".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- smult(hbk[,1:3], monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

spmplot *Interactive scatterplot matrix*

Description

Produces an interactive scatterplot matrix with boxplots or histograms on the main diagonal and possibly robust bivariate contours

Usage

```
spmplot(
  X,
  group,
  plot,
  variables,
  col,
  cex,
  pch,
  labeladd,
  label,
  legend,
  dispopt = c("hist", "box"),
  tag,
  datatooltip,
  databrush,
  trace = FALSE,
  ...
)
```

Arguments

X	data matrix (2D array) containing n observations on p variables or an object of S3 class fsmeda.object returned by fsmult with monitoring=TRUE - a list containing the monitoring of minimum Mahalanobis distance
group	grouping variable. Vector with n elements. Specifies a grouping variable defined as a categorical variable (factor), numeric, or array of strings, or string matrix, and it must have the same number of rows as X. This grouping variable determines the marker and color assigned to each point. Remark: if group is used to distinguish a set of outliers from a set of good units, the id number for the outliers should be the larger (see optional field labeladd of parameter plot for details).
plot	controls the names which are displayed in the margins of the scatter-plot matrix, the labels of the legend the colors and the symbols. If plot is *empty* (plot=FALSE or plot=0 or plot=c() or plot=NULL) empty strings are displayed and no label and no name is added to the plot. If plot=TRUE or plot=1, the names Y1,...,Yp are added to the margins of the the scatter plot matrix else

nothing is added. If plot is a list, it is possible to control not only the names but also, point labels, colors and symbols. More precisely list plot may contain the following elements:

1. labeladd - see parameter labeladd
2. nameY - a character string containing the labels of the variables. As default value, the labels which are added are Y1, ..., Yp. See parameter variables.
3. clr - see parameter col
4. sym - see parameter pch
5. siz - see parameter cex
6. doleg - see parameter legend
7. label - see parameter label

variables — a character string with the names of the variables

col — color specification for the data point. Can be different for each group. By default, the order of the colors is *blue*, *red*, *black*, *magenta*, *green*, *cyan* and *yelow*.

cex — the size of the symbols used for plotting. By default cex=1 the symbol size depends on the number of plots and the size of the figure window. Values larger than 1 will increase the size and values smaller than 1 will decrease the size.

pch — specification of the symbols to use. For example, if there are three groups, and pch=c(1, 3, 4), the first group will be plotted with a circle, the second with a plus, and the third with a 'x' (see ?pch or ?points for a list of symbols. NOTE: not all symbols available in R can be mapped to the symbols in MATLAB.

labeladd — logical, controls wheather the elements belonging to the last group in the scatterplot matrix are labelled with their unit row index or their rowname. The rowname is taken from the parameter label or if it is missing, from the sequence 1:n. The default value is labeladd=FALSE, i.e. no label is added.

label — a character vector of length n (the number of rows in the data matrix) containing the labels of the units. If this field is empty the sequence 1:n will be used to label the units.

legend — logical, controls where a legend is shown or not.

dispopt — controls how to fill the diagonals in the plot (main diagonal of the scatter plot matrix). Set dispopt='hist' (default) to plot histograms, or dispopt='box' to plot boxplots. The style which is used for univariate boxplots is traditional, if the number of groups is less or equal 5, else it is 'compact' (plot boxes using a smaller box style designed for plots with many groups).

tag — Plot handle. String which identifies the handle of the plot which is about to be created. The default is tag='pl_mmd'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.

datatooltip — If datatooltip is not empty the user can use the mouse in order to have information about the unit selected, the step in which the unit enters the search and the associated label. If datatooltip is a list, it is possible to control the aspect of the data cursor (see MATLAB function datacursormode() for more details or see the examples below). The default options are DisplayStyle="Window" and SnapToDataVertex="on".

databrush — Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is TRUE or a list) enables the user to select a set of trajectories in the current plot and to see them highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. In addition, brushed units can be highlighted in the monitoring MD plot. Note that the window style of the other figures is set equal to that which contains the monitoring residual plot. In other words, if the monitoring residual plot is docked all the other figures will be docked too.

If `databrush=TRUE` the default selection tool is a rectangular brush and it is possible to brush only once (that is persist=").

If `databrush=list(...)`, it is possible to use all optional arguments of the MATLAB function `selectdataFS()` and the following optional arguments:

- `persist`: This option can be an empty value or a character containing 'on' or 'off'. The default value is `persist=""`, that is brushing is allowed only once. If `persist="on"` or `persis="off"` brushing can be done as many time as the user requires. If `persist='on'` then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If `persist='off'` every time a new brush is performed units previously brushed are removed.
- `labeladd`: add labels of brushed units in the scatterplot matrix. If this option is '1', we label the units of the last selected group with the unit row index in the matrix X. The default value is labeladd=", i.e. no label is added.

trace — Whether to print intermediate results. Default is `trace=FALSE`.

... — potential further arguments passed to lower level functions.

Value

none

Author(s)

FSDA team, <valentin.todorov@chello.at>

Examples

```
## Not run:
##  Call of spmplot() without optional parameters.
##  Iris data: scatter plot matrix with univariate boxplots on the main
##  diagonal.

X <- iris[,1:4]
group <- iris[,5]
spmplot(X, group, variables=c('SL','SW','PL','PW'), dispopt="box")


##  Example of spmplot() called by routine fsmult().
##  Generate contaminated data.
    n <- 200; p <- 3
```

```
    X <- matrix(rnorm(n*p), ncol=3)
    Xcont <- X
    Xcont[1:5,] <- Xcont[1:5,] + 3

##  spmplot is called automatically by all outlier detection methods, e.g. fsmult()
    out <- fsmult(Xcont, plot=TRUE);

##  Now test the direct use of fsmult(). Set two groups, e.g. those obtained
##  from fsmult().

    group = rep(0, n)
    group[out$outliers] <- 1
##  option 'labeladd' is used to label the outliers
##  By default, the legend identifies the groups with the identifiers
##  given in vector 'group'.
##  Set the colors for the two groups to blue and red.

    spmplot(Xcont, group, col=c("blue", "red"), labeladd=1, dispopt="box")

## End(Not run)
```

`sreg.object` *Description of sreg Objects*

Description

An object of class `sreg.object` holds information about the result of a call to `fsreg`.

Value

The object itself is basically a `list` with the following components:

`beta`	p-by-1 vector containing the estimated regression parameters (in step n-k).
`scale`	scalar containing the estimate of the scale (sigma).
`bs`	p x 1 vector containing the units forming best subset associated with S estimate of regression coefficient.
`residuals`	residuals.
`fittedvalues`	fitted values.
`outliers`	kx1 vector containing the list of the k units declared as outliers or NULL if the sample is homogeneous.
`conflev`	Confidence level which is used to declare units as outliers. Usually `conflev=0.95`, `0.975`, `0.99` (individual alpha) or `conflev=1-0.05/n`, `1-0.025/n`, `1-0.01/n` (simultaneous alpha). Default value is 0.975
`singsub`	Number of subsets wihtout full rank. Notice that `singsub > 0.1*(number of subsamples)` produces a warning

weights	n x 1 vector containing the estimates of the weights
rhofunc	Specifies the rho function which has been used to weight the residuals.
rhofuncparam	Vector which contains the additional parameters for the specified rho function which has been used. For hyperbolic rho function the value of k =sup CVC. For Hampel rho function the parameters a, b and c.
X	the data matrix X
y	the response vector y

The object has class "sreg".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="S"))
    class(out)
    summary(out)

## End(Not run)
```

Sreg_control *Creates an* Sreg_control *object*

Description

Creates an object of class Sreg_control to be used with the fsreg() function, containing various control parameters for calling the MATLAB function Sreg().

Usage

```
Sreg_control(intercept = TRUE, bdp = 0.5,
    rhofunc = c("bisquare", "optimal", "hyperbolic", "hampel", "mdpd", "AS"),
    rhofuncparam, nsamp = 1000, refsteps = 3, reftol = 1e-06,
    refstepsbestr = 50, reftolbestr = 1e-08,
    minsctol = 1e-07, bestr = 5,
    conflev, msg = TRUE, nocheck = FALSE, plot = FALSE)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
bdp	Breakdown point. It measures the fraction of outliers the algorithm should resist. In this case any value greater than 0 but smaller or equal than 0.5 will do fine. Note that given bdp nominal efficiency is automatically determined.
rhofunc	Specifies the rho function which must be used to weight the residuals. Possible values are 'bisquare' 'optimal' 'hyperbolic' 'hampel'.

1. 'bisquare' uses Tukey's rho and psi functions. See TBrho and TBpsi.
2. 'optimal' uses optimal rho and psi functions. See OPTrho and OPTpsi.
3. 'hyperbolic' uses hyperbolic rho and psi functions. See HYPrho and HYPpsi.
4. 'hampel' uses Hampel rho and psi functions. See HArho and HApsi.

The default is 'bisquare'.

`rhofuncparam` Additional parameters for the specified rho function. For hyperbolic rho function it is possible to set up the value of k = sup CVC (the default value of k is 4.5).

For Hampel rho function it is possible to define parameters a, b and c (the default values are a=2, b=4, c=8)

`nsamp` Number of subsamples which will be extracted to find the robust estimator, scalar. If `nsamp=0` all subsets will be extracted. They will be `(n choose p)`. If the number of all possible subset is `<1000` the default is to extract all subsets otherwise just 1000.

`refsteps` Number of refining iterations in each subsample (default is `refsteps=3`). `refsteps = 0` means "raw-subsampling" without iterations.

`reftol` Tolerance for the refining steps. The default value is 1e-6

`refstepsbestr` Scalar defining number of refining iterations for each best subset (default = 50).

`reftolbestr` Tolerance for the refining steps for each of the best subsets. The default value is `reftolbestr=1e-8`.

`minsctol` Value of tolerance for the iterative procedure for finding the minimum value of the scale for each subset and each of the best subsets (It is used by subroutine `minscale.m`). The default value is `minsctol=1e-7`.

`bestr` Defins the number of "best betas" to remember from the subsamples. These will be later iterated until convergence (default is `bestr=5`).

`conflev` Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975

`msg` Controls whether to display or not messages on the screen If `msg==1` (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

`nocheck` Check input arguments, scalar. If `nocheck=TRUE` no check is performed on matrix `y` and matrix `X`. Notice that `y` and `X` are left unchanged. In other words the additional column of ones for the intercept is not added. As default `nocheck=FALSE`.

`plot` Plot on the screen. Scalar. If `plots=TRUE` the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plots=2` the user can also monitor the intermediate plots based on envelope superimposition. If `plots=FALSE` (default) no plot is produced.

Details

Creates an object of class `Sreg_control` to be used with the `fsreg()` function, containing various control parameters.

Value

An object of class "Sreg_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as FSR_control, MMreg_control and LXS_control

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="S", control=Sreg_control(bdp=0.25, nsamp=500)))

## End(Not run)
```

---

sregeda.object *Description of* sregeda *Objects*

Description

An object of class sregeda.object holds information about the result of a call to fsreg when method="S" and monitoring=TRUE.

Value

The object itself is basically a list with the following components:

Beta	matrix containing the S estimator of regression coefficients for each value of bdp.
Scale	vector containing the estimate of the scale (sigma) for each value of bdp. This is the value of the objective function.
BS	p x 1 vector containing the units forming best subset associated with S estimate of regression coefficient.
RES	n x length(bdp) matrix containing the monitoring of scaled residuals for each value of bdp.
Weights	n x length(bdp) matrix containing the estimates of the weights for each value of bdp
Outliers	Boolean matrix containing the list of the units declared as outliers for each value of bdp using confidence level specified in input scalar conflev.
conflev	Confidence level which is used to declare units as outliers. Remark: conflev will be used to draw the horizontal line (confidence band) in the plot.

Singsub	Number of subsets wihtout full rank. Notice that singsub[bdp[jj]] > 0.1*(number of subsamples) produces a warning
rhofunc	Specifies the rho function which has been used to weight the residuals.
rhofuncparam	Vector which contains the additional parameters for the specified rho function which has been used. For hyperbolic rho function the value of k =sup CVC. For Hampel rho function the parameters a, b and c.
X	the data matrix X
y	the response vector y

The object has class "sregeda".

Examples

```
## Not run:
    data(hbk, package="robustbase")
    (out <- fsreg(Y~., data=hbk, method="S", monitoring=TRUE))
    class(out)
    summary(out)

## End(Not run)
```

Sregeda_control *Creates an* Sregeda_control *object*

Description

Creates an object of class Sregeda_control to be used with the fsreg() function, containing various control parameters.

Usage

```
Sregeda_control(intercept = TRUE, bdp = seq(0.5, 0.01, -0.01),
    rhofunc = c("bisquare", "optimal", "hyperbolic", "hampel", "mdpd", "AS"),
    rhofuncparam, nsamp = 1000, refsteps = 3, reftol = 1e-06,
    refstepsbestr = 50, reftolbestr = 1e-08,
    minsctol = 1e-07, bestr = 5,
    conflev, msg = TRUE, nocheck = FALSE, plot = FALSE)
```

Arguments

intercept	Indicator for constant term. Scalar. If intercept=TRUE, a model with constant term will be fitted (default), else, no constant term will be included.
bdp	Breakdown point. It measures the fraction of outliers the algorithm should resist. In this case any value greater than 0 but smaller or equal than 0.5 will do fine. The default value of bdp is a sequence from 0.5 to 0.01 with step 0.01

`rhofunc` Specifies the rho function which must be used to weight the residuals. Possible values are 'bisquare' 'optimal' 'hyperbolic' 'hampel'.

1. 'bisquare' uses Tukey's rho and psi functions. See TBrho and TBpsi.
2. 'optimal' uses optimal rho and psi functions. See OPTrho and OPTpsi.
3. 'hyperbolic' uses hyperbolic rho and psi functions. See HYPrho and HYPpsi.
4. 'hampel' uses Hampel rho and psi functions. See HArho and HApsi.

The default is 'bisquare'.

`rhofuncparam` Additional parameters for the specified rho function. For hyperbolic rho function it is possible to set up the value of k = sup CVC (the default value of k is 4.5).

For Hampel rho function it is possible to define parameters a, b and c (the default values are a=2, b=4, c=8)

`nsamp` Number of subsamples which will be extracted to find the robust estimator, scalar. If `nsamp=0` all subsets will be extracted. They will be `(n choose p)`. If the number of all possible subset is `<1000` the default is to extract all subsets otherwise just 1000.

`refsteps` Number of refining iterations in each subsample (default is `refsteps=3`). `refsteps = 0` means "raw-subsampling" without iterations.

`reftol` Tolerance for the refining steps. The default value is 1e-6

`refstepsbestr` Scalar defining number of refining iterations for each best subset (default = 50).

`reftolbestr` Tolerance for the refining steps for each of the best subsets. The default value is `reftolbestr=1e-8`.

`minsctol` Value of tolerance for the iterative procedure for finding the minimum value of the scale for each subset and each of the best subsets (It is used by subroutine `minscale.m`). The default value is `minsctol=1e-7`.

`bestr` Defins the number of "best betas" to remember from the subsamples. These will be later iterated until convergence (default is `bestr=5`).

`conflev` Confidence level which is used to declare units as outliers. Usually conflev=0.95, 0.975, 0.99 (individual alpha) or conflev=1-0.05/n, 1-0.025/n, 1-0.01/n (simultaneous alpha). Default value is 0.975

`msg` Controls whether to display or not messages on the screen If `msg==1` (default) messages are displayed on the screen about step in which signal took place else no message is displayed on the screen.

`nocheck` Check input arguments, scalar. If `nocheck=TRUE` no check is performed on matrix `y` and matrix `X`. Notice that `y` and `X` are left unchanged. In other words the additional column of ones for the intercept is not added. As default `nocheck=FALSE`.

`plot` Plot on the screen. Scalar. If `plots=TRUE` the plot of minimum deletion residual with envelopes based on n observations and the scatterplot matrix with the outliers highlighted is produced. If `plots=2` the user can also monitor the intermediate plots based on envelope superimposition. If `plots=FALSE` (default) no plot is produced.

Details

Creates an object of class `Sregeda_control` to be used with the `fsreg()` function, containing various control parameters.

Value

An object of class "Sregeda_control" which is basically a list with components the input arguments of the function mapped accordingly to the corresponding Matlab function.

Author(s)

FSDA team

See Also

See Also as FSR_control, MMreg_control and LXS_control

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- fsreg(Y~., data=hbk, method="S", monitoring=TRUE,
    control=Sregeda_control(nsamp=500, rhofunc='hyperbolic')))

## End(Not run)
```

---

summary.fsdalms *Summary Method for* fsdalms *objects*

Description

summary method for class "fsdalms".

Usage

```
## S3 method for class 'fsdalms'
summary(object, correlation = FALSE, ...)
## S3 method for class 'summary.fsdalms'
print(x, digits = max(3, getOption("digits") - 3),
     signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object, x	an object of class "fsdalms" (or "summary.fsdalms"); usually, a result of a call to fsreg.
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
digits	the number of significant digits to use when printing.
signif.stars	logical indicating if "significance stars" should be printer, see printCoefmat.
...	further arguments passed to or from other methods.

Details

summary.fsdalms(), the S3 method, simply returns an (S3) object of class "summary.fsdalms" for which there's a print method:

print.summary.fsdalms prints summary statistics for the forward search (FS) regression estimates. While the function print.fsdalms prints only the robust estimates of the coefficients, print.summary.fsdalms will print also the regression table.

Value

summary.fsdalms returns an summary.fsdalms object, whereas the print methods returns its first argument via invisible, as all print methods do.

See Also

fsreg, summary

Examples

```
## Not run:

    data(Animals, package = "MASS")
    brain <- Animals[c(1:24, 26:25, 27:28),]
    lbrain <- log(brain)
    (fs <- fsreg(brain~body, data=lbrain, method="LTS"))
    summary(fs)

    ## compare to the result of ltsReg() from 'robustbase'
    library(robustbase)
    (lts <- ltsReg(brain~body, data=lbrain))
    summary(lts)

## End(Not run)
```

---

summary.fsdalts *Summary Method for* fsdalts *objects*

Description

summary method for class "fsdalts".

Usage

```
## S3 method for class 'fsdalts'
summary(object, correlation = FALSE, ...)
## S3 method for class 'summary.fsdalts'
print(x, digits = max(3, getOption("digits") - 3),
     signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object, x	an object of class "fsdalts" (or "summary.fsdalts"); usually, a result of a call to fsreg.
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
digits	the number of significant digits to use when printing.
signif.stars	logical indicating if "significance stars" should be printer, see printCoefmat.
...	further arguments passed to or from other methods.

Details

summary.fsdalts(), the S3 method, simply returns an (S3) object of class "summary.fsdalts" for which there's a print method:

print.summary.fsdalts prints summary statistics for the forward search (FS) regression estimates. While the function print.fsdalts prints only the robust estimates of the coefficients, print.summary.fsdalts will print also the regression table.

Value

summary.fsdalts returns an summary.fsdalts object, whereas the print methods returns its first argument via invisible, as all print methods do.

See Also

fsreg, summary

Examples

```
## Not run:

    data(Animals, package = "MASS")
    brain <- Animals[c(1:24, 26:25, 27:28),]
    lbrain <- log(brain)
    (fs <- fsreg(brain~body, data=lbrain, method="LTS"))
    summary(fs)

    ## compare to the result of ltsReg() from 'robustbase'
    library(robustbase)
    (lts <- ltsReg(brain~body, data=lbrain))
    summary(lts)

## End(Not run)
```

summary.fsr *Summary Method for FSR objects*

Description

summary method for class "fsr".

Usage

```
## S3 method for class 'fsr'
summary(object, correlation = FALSE, ...)
## S3 method for class 'summary.fsr'
print(x, digits = max(3, getOption("digits") - 3),
     signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object, x	an object of class "fsr" (or "summary.fsr"); usually, a result of a call to fsreg.
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
digits	the number of significant digits to use when printing.
signif.stars	logical indicating if "significance stars" should be printer, see printCoefmat.
...	further arguments passed to or from other methods.

Details

summary.fsr(), the S3 method, simply returns an (S3) object of class "summary.fsr" for which there's a print method:

print.summary.fsr prints summary statistics for the forward search (FS) regression estimates. While the function print.fsr prints only the robust estimates of the coefficients, print.summary.fsr will print also the regression table.

Value

summary.fsr returns an summary.fsr object, whereas the print methods returns its first argument via invisible, as all print methods do.

See Also

fsreg, summary

Examples

```
## Not run:

    data(Animals, package = "MASS")
    brain <- Animals[c(1:24, 26:25, 27:28),]
    lbrain <- log(brain)
    (fs <- fsreg(brain~body, data=lbrain, method="FS"))
    summary(fs)

## End(Not run)
```

`swissbanknotes` *Swiss banknote data*

Description

Six variables measured on 100 genuine and 100 counterfeit old (printed before the second world war) Swiss 1000-franc bank notes (Flury and Riedwyl, 1988).

Usage

```
data(swissbanknotes)
```

Format

A data frame with 200 observations on the following 7 variables.

`length` Length of bill, mm

`left` Width of left edge, mm

`right` Width of right edge, mm

`bottom` Bottom margin width, mm

`top` Top margin width, mm

`diagonal` Length of image diagonal, mm

`class` 1 = genuine, 2 = counterfeit

Source

Flury, B. and Riedwyl, H. (1988). *Multivariate Statistics: A practical approach.* London: Chapman & Hall.

References

Weisberg, S. (2005). *Applied Linear Regression*, 3rd edition. New York: Wiley, Problem 12.5.

Examples

```
library(rrcov)
data(swissbanknotes)
head(swissbanknotes)
plot(CovMcd(swissbanknotes[, 1:6]), which="pairs", col=swissbanknotes$class)
```

`swissheads` *Swiss heads data*

Description

Six dimensions in millimetres of the heads of 200 twenty year old Swiss soldiers (Flury and Riedwyl, 1988, p. 218 and also Flury, 1997, p. 6).

The data were collected to determine the variability in size and shape of heads of young men in order to help in the design of a new protection mask for the Swiss army.

Usage

```
data(swissheads)
```

Format

A data frame with 200 observations on the following 6 variables.

`minimal_frontal_breadth` Minimal frontal breadth, mm

`breadth_angulus_mandibulae` Breadth of angulus mandibulae, mm

`true_facial_height` True facial height, mm

`length_glabella_nasi` Length from glabella to apex nasi, mm

`length_tragion_nasion` Length from tragion to nasion, mm

`length_tragion_gnathion` Length from tragion to gnathion, mm

Source

Flury, B. and Riedwyl, H. (1988). *Multivariate Statistics: A practical approach.* London: Chapman & Hall.

References

Atkinson, A. C., Riani, M. and Cerioli, A. (2004) *Exploring multivariate data with the forward search*, New York: Springer-Verlag.

Examples

```
library(rrcov)
data(swissheads)
head(swissheads)
plot(CovMcd(swissheads), which="pairs")
```

tclusteda.object *Objects returned by the function* tclustfsda *with the option* monitoring=TRUE

Description

An object of class tclusteda.object holds information about the result of a call to tclustfsda with the option monitoring=TRUE.

Value

The functions print() and summary() are used to obtain and print a summary of the results. An object of class tclusteda is a list containing at least the following components:

call	the matched call
k	number of groups
alpha	trimming level
restrfactor	restriction factor
IDX	an n-by-length(alpha) vector containing assignment of each unit to each of the k groups. Cluster names are integer numbers from 1 to k. 0 indicates trimmed observations. The first column refers to alpha[1], the second column refers to alpha[2], ..., the last column refers to alpha[length(alpha)].
MU	a 3 dimensional array of size k-by-p-by-length(alpha) containing the monitoring of the centroid for each value of alpha. MU[,,1], refers to alpha[1] ..., MU(,,length(alpha)] refers to alpha[length(alpha)]. The first row in each slice refers to group 1, second row refers to group 2, etc.
SIGMA	A list of length length(alpha) containing in element j, with j=1, 2, ..., length(alpha), the 3D array of size p-by-p-by-k containing the k (constrained) estimated covariance matrices associated with alpha[j].
Amon	Amon stands for alpha monitoring. Matrix of size (length(alpha)-1)-by-4 which contains for two consecutive values of alpha the monitoring of three quantities (change in classification, change in centroid location, change in covariance location). • 1st col = value of alpha. • 2nd col = ARI index. • 3rd col = squared Euclidean distance between centroids. • 4th col = squared Euclidean distance between covariance matrices.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- tclustfsda(hbk[, 1:3], k=2, monitoring=TRUE))
class(out)
summary(out)

## End(Not run)
```

`tclustfsda` *Computes trimmed clustering with scatter restrictions*

Description

Partitions the points in the n-by-v data matrix `Y` into `k` clusters. This partition minimizes the trimmed sum, over all clusters, of the within-cluster sums of point-to-cluster-centroid distances. Rows of Y correspond to points, columns correspond to variables. Returns in the output object of class `tclustfsda.object` an n-by-1 vector `idx` containing the cluster indices of each point. By default, `tclustfsda()` uses (squared), possibly constrained, Mahalanobis distances.

Usage

```
tclustfsda(
  x,
  k,
  alpha,
  restrfactor = 12,
  monitoring = FALSE,
  plot = FALSE,
  nsamp,
  refsteps = 15,
  reftol = 1e-13,
  equalweights = FALSE,
  mixt = 0,
  msg = FALSE,
  nocheck = FALSE,
  startv1 = 1,
  RandNumbForNini,
  restrtype = c("eigen", "deter"),
  UnitsSameGroup,
  numpool,
  cleanpool,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
k	Number of groups.

alpha — A scalar between 0 and 0.5 or an integer specifying the number of observations which have to be trimmed. If alpha=0, tclust reduces to traditional model based or mixture clustering (mclust): see for example the Matlab function gmdistribution.

More in detail, if 0 < alpha < 1 clustering is based on h = floor(n * (1-alpha)) observations, else if alpha is an integer greater than 1 clustering is based on h = n - floor(alpha). If monitoring=TRUE, alpha is a vector which specifies the values of trimming levels which have to be considered - contains decresing elements which lie in the interval 0 and 0.5. For example if alpha=c(0.1, 0.05, 0), tclust() considers these 3 values of trimming level. The default for alpha is vector alpha=c(0.1, 0.05, 0). The sequence is forced to be monotonically decreasing.

restrfactor — Positive scalar which constrains the allowed differences among group scatters. Larger values imply larger differences of group scatters. On the other hand a value of 1 specifies the strongest restriction forcing all eigenvalues/determinants to be equal and so the method looks for similarly scattered (respectively spherical) clusters. The default is to apply restrfactor to eigenvalues. In order to apply restrfactor to determinants it is necessary to use the optional input argument restrtype.

monitoring — If monitoring=TRUE TCLUST is performed for a series of values of the trimming factor alpha given k (number of groups) and given c (restriction factor).

plot — If plot=FALSE (default) or plot=0 no plot is produced. If plot=TRUE and monitoring=FALSE a plot with the classification is shown (using the spmplot function). The plot can be:

- for p = 1, a histogram of the univariate data,
- for p = 2, a bivariate scatterplot,
- for p > 2, a scatterplot matrix generated by the MATLAB function spmplot().

When p >= 2 the following additional features are offered (for p = 1 the behaviour is forced to be as for plots=TRUE):

- plot = 'contourf' adds in the background of the bivariate scatterplots a filled contour plot. The colormap of the filled contour is based on grey levels as default. This argument may also be inserted in a field named 'type' of a list. In the latter case it is possible to specify the additional field 'cmap', which changes the default colors of the color map used. The field 'cmap' may be a three-column matrix of values in the range [0,1] where each row is an RGB triplet that defines one color. Check the colormap function for additional informations.
- plot = 'contour' adds in the background of the bivariate scatterplots a contour plot. The colormap of the contour is based on grey levels as default. This argument may also be inserted in a field named type of a list. In the latter case it is possible to specify the additional field cmap, which changes the default colors of the color map used. The field cmap may be a three-column matrix of values in the range [0,1] where each row is an RGB triplet that defines one color. Check the colormap() (MATLAB) function for additional information.

- plot = 'ellipse' superimposes confidence ellipses to each group in the bivariate scatterplots. The size of the ellipse is qchisq(0.95, 2), i.e. the confidence level used by default is 95 percent. This argument may also be inserted in a field named type of a list. In the latter case it is possible to specify the additional field conflev, which specifies the confidence level to use and it is a value between 0 and 1.
- plot = 'boxplotb' superimposes on the bivariate scatterplots the bivariate boxplots for each group, using the boxplotb function. This argument may also be inserted in a field named type of a list.

The parameter plot can be also a list and in this case its elements are:

- type - specifies the type of plot as when plot option is a character. Therefore, plots$type can be one of 'contourf', 'contour', 'ellipse' or 'boxplotb'.
- cmap - used to set a colormap for the plot type (MATLAB style). For example, plot$cmap = 'autumn'. See the MATLAB help of colormap for a list of colormap possiblilites.
- conflev - this is the confidence level for the confidence ellipses. It must me a scalar between 0 and 1.

If plot=TRUE and monitoring=TRUE two plots are shown. The first plot (*monitor plot*) shows three panels with the monitoring between two consecutive values of alpha: (i) the change in classification using ARI index (top panel), (ii) the change in centroids using squared euclidean distances (central panel) and (iii) the change in covariance matrices using squared euclidean distance (bottom panel).

The second plot (*gscatter plot*) shows a series of subplots which monitor the classification for each value of alpha. In order to make sure that consistent labels are used for the groups, between two consecutive values of alpha, we assign label r to a group if this group shows the smallest distance with group r for the previous value of alpha. The type of plot which is used to monitor the stability of the classification depends on the data dimensionality p.

- for p = 1, a histogram of the univariate data (the MATLAB function histFS() is called),
- for p = 2, a bivariate scatterplot (the MATLAB function gscatter() is called),
- for p > 2, a scatterplot of the first two principal components (function gscatter() is called and we show on the axes titles the percentage of variance explained by the first two principal components).

Also in the case of monitoring=TRUE the parameter plot can be a list and its elements are:

- name: character vector which enables to specify which plot to display. name = "gscatter" produces a figure with a series of subplots which show the classification for each value of alpha. name = "monitor" shows a figure with three panels which monitor between two consecutive values of alpha the change in classification using ARI index (top panel), the change in centroids using squared euclidean distances (central panel), the change in covariance matrices using squared euclidean distance (bottom panel). If this field is not specified, by default name=c("gscatter", "monitor") and both figures will be shown.

- alphasel: a numeric vector which specifies for which values of alpha it is possible to see the classification. For example if `alphasel = c(0.05, 0.02)`, the classification will be shown just for `alpha=0.05` and `alpha=0.02`. If this field is not specified `alphasel=alpha` and therefore the classification is shown for each value of alpha.

`nsamp` If a scalar, it contains the number of subsamples which will be extracted. If `nsamp = 0` all subsets will be extracted. Remark - if the number of all possible subset is greater than 300 the default is to extract all subsets, otherwise just 300. If `nsamp` is a matrix it contains in the rows the indexes of the subsets which have to be extracted. `nsamp` in this case can be conveniently generated by function `subsets()`. `nsamp` can have `k` columns or `k * (p + 1)` columns. If `nsamp` has `k` columns the `k` initial centroids each iteration i are given by `X[nsamp[i,] ,]` and the covariance matrices are equal to the identity.

If `nsamp` has `k * (p + 1)` columns, the initial centroids and covariance matrices in iteration `i` are computed as follows:

- X1 <- X[nsamp[i ,] ,]
- mean(X1[1:p + 1,]) contains the initial centroid for group 1
- cov(X1[1:p + 1,]) contains the initial cov matrix for group 1
- mean(X1[(p + 2):(2*p + 2),]) contains the initial centroid for group 2
- cov(X1[(p + 2):(2*p + 2),]) contains the initial cov matrix for group 2
- ...
- mean(X1[(k-1)*p+1):(k*(p+1),]) contains the initial centroids for group k
- cov(X1[(k-1)*p+1):(k*(p+1),]) contains the initial cov matrix for group k.

REMARK: If `nsamp` is not a scalar, the option `startv1` given below is ignored. More precisely, if `nsamp` has `k` columns `startv1 = 0` else if `nsamp` has `k*(p+1)` columns option `startv1=1`.

`refsteps` Number of refining iterations in each subsample. Default is `refsteps=15`. `refsteps = 0` means "raw-subsampling" without iterations.

`reftol` Tolerance of the refining steps. The default value is 1e-14

`equalweights` A logical specifying wheather cluster weights in the concentration and assignment steps shall be considered. If `equalweights=TRUE` we are (ideally) assuming equally sized groups, else if `equalweights = false` (default) we allow for different group weights. Please, check in the given references which functions are maximized in both cases.

`mixt` Specifies whether mixture modelling or crisp assignment approach to model based clustering must be used. In the case of mixture modelling parameter mixt also controls which is the criterion to find the untrimmed units in each step of the maximization. If `mixt >=1` mixture modelling is assumed else crisp assignment. The default value is `mixt=0`, i.e. crisp assignment. Please see for details the provided references. The parameter `mixt` also controls the criterion to select the units to trim. If `mixt = 2` the `h` units are those which give the largest contribution to the likelihood, else if `mixt=1` the criterion to select the `h` units is exactly the same as the one which is used in crisp assignment.

`msg` Controls whether to display or not messages on the screen. If `msg==TRUE` messages are displayed on the screen. If `msg=2`, detailed messages are displayed, for example the information at iteration level.

nocheck — Check input arguments. If nocheck=TRUE no check is performed on matrix X. The default nocheck=FALSE.

startv1 — How to initialize centroids and covariance matrices. Scalar. If startv1=1 then initial centroids and covariance matrices are based on (p+1) observations randomly chosen, else each centroid is initialized taking a random row of input data matrix and covariance matrices are initialized with identity matrices. The default value isstartv1=1.

Remark 1: in order to start with a routine which is in the required parameter space, eigenvalue restrictions are immediately applied.

Remark 2 - option startv1 is used just if nsamp is a scalar (see for more details the help associated with nsamp).

RandNumbForNini

pre-extracted random numbers to initialize proportions. Matrix of size k-by-nrow(nsamp) containing the random numbers which are used to initialize the proportions of the groups. This option is effective just if nsamp is a matrix which contains pre-extracted subsamples. The purpose of this option is to enable to user to replicate the results in case routine tclustreg*() is called using a parfor instruction (as it happens for example in routine IC, where tclustreg() is called through a parfor for different values of the restriction factor). The default is that RandNumbForNini is empty - then uniform random numbers are used.

restrtype — Type of restriction to be applied on the cluster scatter matrices. Valid values are 'eigen' (default), or 'deter'. "eigen" implies restriction on the eigenvalues while "deter" implies restriction on the determinants.

UnitsSameGroup — List of the units which must (whenever possible) have a particular label. For example UnitsSameGroup=c(20, 26), means that group which contains unit 20 is always labelled with number 1. Similarly, the group which contains unit 26 is always labelled with number 2, (unless it is found that unit 26 already belongs to group 1). In general, group which contains unit UnitsSameGroup(r) where r=2, ...length(kk)-1 is labelled with number r (unless it is found that unit UnitsSameGroup(r) has already been assigned to groups 1, 2, ..., r-1.

numpool — The number of parallel sessions to open. If numpool is not defined, then it is set equal to the number of physical cores in the computer.

cleanpool — Logical, indicating if the open pool must be closed or not. It is useful to leave it open if there are subsequent parallel sessions to execute, so that to save the time required to open a new pool.

trace — Whether to print intermediate results. Default is trace=FALSE.

... — potential further arguments passed to lower level functions.

Details

This iterative algorithm initializes k clusters randomly and performs concentration steps in order to improve the current cluster assignment. The number of maximum concentration steps to be performed is given by input parameter refsteps. For approximately obtaining the global optimum, the system is initialized nsamp times and concentration steps are performed until convergence or refsteps is reached. When processing more complex data sets higher values of nsamp

and refsteps have to be specified (obviously implying extra computation time). However, if more then 10 per cent of the iterations do not converge, a warning message is issued, indicating that nsamp has to be increased.

Value

Depending on the input parameter monitoring, one of the following objects will be returned:

1. tclustfsda.object
2. tclusteda.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Garcia-Escudero, L.A., Gordaliza, A., Matran, C. and Mayo-Iscar, A. (2008). A General Trimming Approach to Robust Cluster Analysis. Annals of Statistics, Vol. 36, 1324-1345. doi:10.1214/07-AOS515.

Examples

```
## Not run:

data(hbk, package="robustbase")
(out <- tclustfsda(hbk[, 1:3], k=2))
class(out)
summary(out)

##  TCLUST of Gayser data with three groups (k=3), 10%% trimming (alpha=0.1)
##      and restriction factor (c=10000)
data(geyser2)
(out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000))

## Use the plot options to produce more complex plots ----------

##  Plot with all default options
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, plot=TRUE)

##  Default confidence ellipses.
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, plot="ellipse")

##  Confidence ellipses specified by the user: confidence ellipses set to 0.5
plots <- list(type="ellipse", conflev=0.5)
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, plot=plots)

##  Confidence ellipses set to 0.9
plots <- list(type="ellipse", conflev=0.9)
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, plot=plots)

##  Contour plots
```

```
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, plot="contour")

##  Filled contour plots with additional options: contourf plot with a named colormap.
##  Here we define four MATLAB-like colormaps, but the user can define anything else,
##  presented by a matrix with three columns which are the RGB triplets.

summer <- as.matrix(data.frame(x1=seq(from=0, to=1, length=65),
                               x2=seq(from=0.5, to=1, length=65),
                               x3=rep(0.4, 65)))
spring <- as.matrix(data.frame(x1=rep(1, 65),
                               x2=seq(from=0, to=1, length=65),
                               x3=seq(from=1, to=0, length=65)))
winter <- as.matrix(data.frame(x1=rep(0, 65),
                               x2=seq(from=0, to=1, length=65),
                               x3=seq(from=1, to=0, length=65)))
autumn <- as.matrix(data.frame(x1=rep(1, 65),
                               x2=seq(from=0, to=1, length=65),
                               x3=rep(0, 65)))

out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000,
      plot=list(type="contourf", cmap=autumn))
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000,
      plot=list(type="contourf", cmap=winter))
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000,
      plot=list(type="contourf", cmap=spring))
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000,
      plot=list(type="contourf", cmap=summer))


##  We compare the output using three different values of restriction factor
##      nsamp is the number of subsamples which will be extracted
data(geyser2)
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10000, nsamp=500, plot="ellipse")
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=10, nsamp=500, refsteps=10, plot="ellipse")
out <- tclustfsda(geyser2, k=3, alpha=0.1, restrfactor=1, nsamp=500, refsteps=10, plot="ellipse")

##  TCLUST applied to M5 data: A bivariate data set obtained from three normal
##  bivariate distributions with different scales and proportions 1:2:2. One of the
##  components is very overlapped with another one. A 10 per cent background noise is
##  added uniformly distributed in a rectangle containing the three normal components
##  and not very overlapped with the three mixture components. A precise description
##  of the M5 data set can be found in Garcia-Escudero et al. (2008).
##

data(M5data)
plot(M5data[, 1:2])

##  Scatter plot matrix
library(rrcov)
plot(CovClassic(M5data[,1:2]), which="pairs")

out <- tclustfsda(M5data[,1:2], k=3, alpha=0, restrfactor=1000, nsamp=100, plot=TRUE)
out <- tclustfsda(M5data[,1:2], k=3, alpha=0, restrfactor=10, nsamp=100, plot=TRUE)
```

```
out <- tclustfsda(M5data[,1:2], k=3, alpha=0.1, restrfactor=1, nsamp=1000,
        plot=TRUE, equalweights=TRUE)
out <- tclustfsda(M5data[,1:2], k=3, alpha=0.1, restrfactor=1000, nsamp=100, plot=TRUE)

##  TCLUST with simulated data: 5 groups and 5 variables
##
n1 <- 100
n2 <- 80
n3 <- 50
n4 <- 80
n5 <- 70
p <- 5
Y1 <- matrix(rnorm(n1 * p) + 5, ncol=p)
Y2 <- matrix(rnorm(n2 * p) + 3, ncol=p)
Y3 <- matrix(rnorm(n3 * p) - 2, ncol=p)
Y4 <- matrix(rnorm(n4 * p) + 2, ncol=p)
Y5 <- matrix(rnorm(n5 * p), ncol=p)

group <- c(rep(1, n1), rep(2, n2), rep(3, n3), rep(4, n4), rep(5, n5))
Y <- Y1
Y <- rbind(Y, Y2)
Y <- rbind(Y, Y3)
Y <- rbind(Y, Y4)
Y <- rbind(Y, Y5)
dim(Y)
table(group)
out <- tclustfsda(Y, k=5, alpha=0.05, restrfactor=1.3, refsteps=20, plot=TRUE)

##  Automatic choice of best number of groups for Geyser data ------------------------
##
data(geyser2)
maxk <- 6
CLACLA <- matrix(0, nrow=maxk, ncol=2)
CLACLA[,1] <- 1:maxk
MIXCLA <- MIXMIX <- CLACLA

for(j in 1:maxk) {
    out <- tclustfsda(geyser2, k=j, alpha=0.1, restrfactor=5)
    CLACLA[j, 2] <- out$CLACLA
}

for(j in 1:maxk) {
    out <- tclustfsda(geyser2, k=j, alpha=0.1, restrfactor=5, mixt=2)
    MIXMIX[j ,2] <- out$MIXMIX
    MIXCLA[j, 2] <- out$MIXCLA
}

oldpar <- par(mfrow=c(1,3))
plot(CLACLA[,1:2], type="l", xlim=c(1, maxk), xlab="Number of groups", ylab="CLACLA")
plot(MIXMIX[,1:2], type="l", xlim=c(1, maxk), xlab="Number of groups", ylab="MIXMIX")
plot(MIXCLA[,1:2], type="l", xlim=c(1, maxk), xlab="Number of groups", ylab="MIXCLA")
par(oldpar)
```

```
##  Monitoring examples ----------------------------------------

##  Monitoring using Geyser data

##  Monitoring using Geyser data (all default options)
##  alpha and restriction factor are not specified therefore alpha=c(0.10, 0.05, 0)
##  is used while the restriction factor is set to c=12
out <- tclustfsda(geyser2, k=3, monitoring=TRUE)

##  Monitoring using Geyser data with alpha and c specified.
out <- tclustfsda(geyser2, k=3, restrfac=100, alpha=seq(0.10, 0, by=-0.01), monitoring=TRUE)

##  Monitoring using Geyser data with plot argument specified as a list.
##      The trimming levels to consider in this case are 31 values of alpha
##
out <- tclustfsda(geyser2, k=3, restrfac=100, alpha=seq(0.30, 0, by=-0.01), monitoring=TRUE,
        plot=list(alphasel=c(0.2, 0.10, 0.05, 0.01)), trace=TRUE)

##  Monitoring using Geyser data with argument UnitsSameGroup
##
##     Make sure that group containing unit 10 is in a group which is labelled "group 1"
##      and group containing unit 12 is in group which is labelled "group 2"
##
##      Mixture model is used (mixt=2)
##
out <- tclustfsda(geyser2, k=3, restrfac=100, alpha=seq(0.30, 0, by=-0.01), monitoring=TRUE,
        mixt=2, UnitsSameGroup=c(10, 12), trace=TRUE)

##  Monitoring using M5 data
data(M5data)

##  alphavec=vector which contains the trimming levels to consider
##  in this case 31 values of alpha are considered
alphavec <- seq(0.10, 0, by=-0.02)
out <- tclustfsda(M5data[, 1:2], 3, alpha=alphavec, restrfac=1000, monitoring=TRUE,
    nsamp=1000, plots=TRUE)

## End(Not run)
```

`tclustfsda.object` *Objects returned by the function* `tclustfsda`

Description

An object of class `tclustfsda.object` holds information about the result of a call to `tclustfsda`.

Value

The functions `print()` and `summary()` are used to obtain and print a summary of the results. An object of class `tclustfsda` is a list containing at least the following components:

call	the matched call
muopt	a k-by-p matrix containing cluster centroid locations. Robust estimate of final centroids of the groups
sigmaopt	a p-by-p-by-k array rray containing estimated constrained covariance for the k groups
idx	a vector of length n containing assignment of each unit to each of the k groups. Cluster names are integer numbers from 1 to k. 0 indicates trimmed observations.
size	a matrix of size (k+1)-by-3. The 1st col is sequence from 0 to k (cluster name); the 2nd col is the number of observations in each cluster; the 3rd col is the percentage of observations in each cluster. Remark: 0 denotes unassigned units.
postprob	n-by-k matrix containing posterior probabilities. postprob[i, j] contains posterior probabilitiy of unit i from component (cluster) j. For the trimmed units posterior probabilities are 0.
emp	"Empirical" statistics computed on final classification. When convergence is reached, emp=0. When convergence is not obtained, this field is a list which contains the statistics of interest: idxemp (ordered from 0 to k*, k* being the number of groups with at least one observation and 0 representing the possible group of outliers), muemp, sigmaemp and sizemp, which are the empirical counterparts of idx, muopt, sigmaopt and size.
MIXMIX	BIC which uses parameters estimated using the mixture loglikelihood and the maximized mixture likelihood as goodness of fit measure. Remark: this output is present only if mixt=TRUE.
MIXCLA	BIC which uses parameters estimated using the mixture loglikelihood and the maximized mixture likelihood as goodness of fit measure. Remark: this output is present only if mixt=TRUE.
CLACLA	BIC which uses the classification likelihood based on parameters estimated using the classification likelihood. Remark: this output is present only if mixt=FALSE.
NlogL	-2 * classification log likelihood. In the case of full convergence -NlogL/2 == obj.
notconver	number of subsets without convergence
bs	a vector of length k containing the units forming initial subset associated with muopt.
obj	value of the objective function which is minimized (value of the best returned solution).
equalweights	if equalweights=TRUE means that in the clustering procedure we (ideally) assumed equal cluster weights else (equalweitghts=FALSE means that we allowed for different cluster sizes.
h	number of observations that have determined the centroids (number of untrimmed units).
fullsol	a vector of size nsamp which contains the value of the objective function at the end of the iterative process for each extracted subsample.
X	the original data matrix X.

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- tclustfsda(hbk[, 1:3], k=2))
class(out)
summary(out)

## End(Not run)
```

`tclustIC` *Performs cluster analysis by calling* `tclustfsda` *for different number of groups* `k` *and restriction factors* `c`

Description

Computes the values of BIC (MIXMIX), ICL (MIXCLA) or CLA (CLACLA), for different values of `k` (number of groups) and different values of `c` (restriction factor), for a prespecified level of trimming (the last two letters in the name stand for 'Information Criterion'). In order to minimize randomness, given `k`, the same subsets are used for each value of `c`.

Usage

```
tclustIC(
  x,
  kk = 1:5,
  cc = c(1, 2, 4, 8, 16, 32, 64, 128),
  alpha = 0,
  whichIC = c("ALL", "MIXMIX", "MIXCLA", "CLACLA"),
  nsamp,
  refsteps = 15,
  reftol = 1e-14,
  equalweights = FALSE,
  msg = TRUE,
  nocheck = FALSE,
  plot = FALSE,
  startv1 = 1,
  restrtype = c("eigen", "deter"),
  UnitsSameGroup,
  numpool,
  cleanpool,
  trace = FALSE,
  ...
)
```

Arguments

x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
kk	an integer vector specifying the number of mixture components (clusters) for which the information criteria are be calculated. By default `kk=1:5`.
cc	an vector specifying the values of the restriction factor which have to be considered. By default `cc=c(1, 2, 4, 8, 16, 32, 64, 128)`.
alpha	Global trimming level. A scalar between 0 and 0.5 or an integer specifying the number of observations which have to be trimmed. If `alpha=0` all observations are considered. By default `alpha=0`. More in detail, if `0 < alpha < 1` clustering is based on `h = fix(n * (1-alpha))` observations, else if alpha is an integer greater than 1 clustering is based on `h = n - floor(alpha)`.
whichIC	A character value which specifies which information criteria must be computed for each k (number of groups) and each value of the restriction factor c. Possible values for `whichIC` are: • "MIXMIX": a mixture model is fitted and for computing the information criterion the mixture likelihood is used. This option corresponds to the use of the Bayesian Information criterion (BIC). In output just the matrix `MIXMIX` is given. • "MIXCLA": a mixture model is fitted but to compute the information criterion the classification likelihood is used. This option corresponds to the use of the Integrated Complete Likelihood (ICL). In the output just the matrix `MIXCLA` is given. • "CLACLA": everything is based on the classification likelihood. This information criterion will be called CLA. In the output just the matrix `CLACLA` is given. • "ALL": both classification and mixture likelihood are used. In this case all three information criteria CLA, ICL and BIC are computed. In the output all three matrices `MIXMIX`, `MIXCLA` and `CLACLA` are given.
nsamp	If a scalar, it contains the number of subsamples which will be extracted. If `nsamp = 0` all subsets will be extracted. Remark - if the number of all possible subset is greater than 300 the default is to extract all subsets, otherwise just 300. If `nsamp` is a matrix it contains in the rows the indexes of the subsets which have to be extracted. `nsamp` in this case can be conveniently generated by function `subsets()`. `nsamp` can have k columns or `k * (p + 1)` columns. If `nsamp` has k columns the k initial centroids each iteration i are given by `X[nsamp[i,] ,]` and the covariance matrices are equal to the identity. If `nsamp` has `k * (p + 1)` columns, the initial centroids and covariance matrices in iteration `i` are computed as follows: • X1 <- X[nsamp[i ,] ,] • mean(X1[1:p + 1,]) contains the initial centroid for group 1 • cov(X1[1:p + 1,]) contains the initial cov matrix for group 1

- mean(X1[(p + 2):(2*p + 2),]) contains the initial centroid for group 2
- cov(X1[(p + 2):(2*p + 2),]) contains the initial cov matrix for group 2
- ...
- mean(X1[(k-1)*p+1):(k*(p+1),]) contains the initial centroids for group k
- cov(X1[(k-1)*p+1):(k*(p+1),]) contains the initial cov matrix for group k.

REMARK: If `nsamp` is not a scalar, the option `startv1` given below is ignored. More precisely, if `nsamp` has `k` columns `startv1 = 0` else if `nsamp` has `k*(p+1)` columns option `startv1=1`.

`refsteps` Number of refining iterations in each subsample. Default is `refsteps=15`. `refsteps = 0` means "raw-subsampling" without iterations.

`reftol` Tolerance of the refining steps. The default value is 1e-14

`equalweights` A logical specifying wheather cluster weights in the concentration and assignment steps shall be considered. If `equalweights=TRUE` we are (ideally) assuming equally sized groups, else if `equalweights = false` (default) we allow for different group weights. Please, check in the given references which functions are maximized in both cases.

`msg` Controls whether to display or not messages on the screen If `msg==TRUE` (default) messages are displayed on the screen. If `msg=2`, detailed messages are displayed, for example the information at iteration level.

`nocheck` Check input arguments. If `nocheck=TRUE` no check is performed on matrix `X`. The default `nocheck=FALSE`.

`plot` If `plot=TRUE`, a plot of the BIC (MIXMIX), ICL (MIXCLA) curve and CLACLA is shown on the screen. The plots which are shown depend on the input option `whichIC`.

`startv1` How to initialize centroids and covariance matrices. Scalar. If `startv1=1` then initial centroids and covariance matrices are based on `(p+1)` observations randomly chosen, else each centroid is initialized taking a random row of input data matrix and covariance matrices are initialized with identity matrices. The default value is`startv1=1`.

Remark 1: in order to start with a routine which is in the required parameter space, eigenvalue restrictions are immediately applied.

Remark 2 - option `startv1` is used only if `nsamp` is a scalar (for more details see the help associated with `nsamp`).

`restrtype` Type of restriction to be applied on the cluster scatter matrices. Valid values are `'eigen'` (default), or `'deter'`. `"eigen"` implies restriction on the eigenvalues while `"deter"` implies restriction on the determinants.

`UnitsSameGroup` List of the units which must (whenever possible) have a particular label. For example `UnitsSameGroup=c(20, 26)`, means that group which contains unit 20 is always labelled with number 1. Similarly, the group which contains unit 26 is always labelled with number 2, (unless it is found that unit 26 already belongs to group 1). In general, group which contains unit `UnitsSameGroup(r)` where `r=2, ...length(kk)-1` is labelled with number `r` (unless it is found that unit `UnitsSameGroup(r)` has already been assigned to groups `1, 2, ..., r-1`. The default value of `UnitsSameGroup` is an empty list, that is consistent labels are not imposed.

numpool	The number of parallel sessions to open. If numpool is not defined, then it is set equal to the number of physical cores in the computer.
cleanpool	Logical, indicating if the open pool must be closed or not. It is useful to leave it open if there are subsequent parallel sessions to execute, so that to save the time required to open a new pool.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

An S3 object of class tclustic.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Cerioli, A., Garcia-Escudero, L.A., Mayo-Iscar, A. and Riani M. (2017). Finding the Number of Groups in Model-Based Clustering via Constrained Likelihoods, *Journal of Computational and Graphical Statistics*, pp. 404-416, https://doi.org/10.1080/10618600.2017.1390469.

See Also

tclustfsda, tclustICplot, tclustICsol, carbikeplot

Examples

```
 ## Not run:
 data(geyser2)
 (out <- tclustIC(geyser2, whichIC="MIXMIX", plot=FALSE, alpha=0.1))
 summary(out)

## End(Not run)

 ## Not run:
 data(flea)
 Y <- as.matrix(flea[, 1:(ncol(flea)-1)])    # select only the numeric variables
 rownames(Y) <- 1:nrow(Y)
 head(Y)

 (out <- tclustIC(Y, whichIC="CLACLA", plot=FALSE, alpha=0.1, nsamp=100, numpool=1))
 summary(out)

## End(Not run)
```

tclustic.object *Objects returned by the function* tclustIC

Description

An object of class tclustic.object holds information about the result of a call to tclustIC.

Value

The functions print() and summary() are used to obtain and print a summary of the results. An object of class tclustic is a list containing at least the following components:

call	the matched call
kk	a vector containing the values of k (number of components) which have been considered. This vector is identical to the optional argument kk (default is kk=1:5.
cc	a vector containing the values of c (values of the restriction factor) which have been considered. This vector is identical to the optional argument cc (defalt is cc=c(1, 2, 4, 8, 16, 32, 64, 128).
alpha	trimming level
whichIC	Information criteria used
CLACLA	a matrix of size length(kk)-times-length(cc) containinig the value of the penalized classification likelihood. This output is present only if whichIC="CLACLA" or whichIC="ALL".
IDXCLA	a matrix of lists of size length(kk)-times-length(cc) containinig the assignment of each unit using the classification model. This output is present only if whichIC="CLACLA" or whichIC="ALL".
MIXMIX	a matrix of size length(kk)-times-length(cc) containinig the value of the penalized mixtrue likelihood. This output is present only if whichIC="MIXMIX" or whichIC="ALL".
IDXMIX	a matrix of lists of size length(kk)-times-length(cc) containinig the assignment of each unit using the classification model. This output is present only if whichIC="MIXMIX" or whichIC="ALL".
MIXCLA	a matrix of size length(kk)-times-length(cc) containinig the value of the ICL criterion. This output is present only if whichIC="MIXCLA" or whichIC="ALL".

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- tclustIC(hbk[, 1:3]))
class(out)
summary(out)

## End(Not run)
```

`tclustICplot` *Plots information criterion as a function of* `c` *and* `k`*, based on the solutions obtained by* `tclustIC`

Description

The function `tclustICplot()` takes as input an object of class `tclustic.object`, the output of function `tclustIC` (that is a series of matrices which contain the values of the information criteria BIC/ICL/CLA for different values of `k` and `c`) and plots them as function of `c` or of `k`. The plot enables interaction in the sense that, if option `databrush` has been activated, it is possible to click on a point in the plot and to see the associated classification in the scatter plot matrix.

Usage

```
tclustICplot(
  out,
  whichIC = c("ALL", "MIXMIX", "MIXCLA", "CLACLA"),
  tag,
  datatooltip,
  databrush,
  nameY,
  trace = FALSE,
  ...
)
```

Arguments

`out`	An S3 object of class `tclustic.object` (output of `tclustIC`) containing the values of the information criteria BIC (MIXMIX), ICL (MIXCLA) or CLA (CLACLA), for different values of k (number of groups) and different values of c (restriction factor), for a prespecified level of trimming.
`whichIC`	Specifies the information criterion to use for the plot. See `tclustIC()` for the possible values of whichIC.
`tag`	plot handle. String which identifies the handle of the plot which is about to be created. The default is to use tag 'pl_IC'. Notice that if the program finds a plot which has a tag equal to the one specified by the user, then the output of the new plot overwrites the existing one in the same window else a new window is created.
`datatooltip`	Interactive clicking. It is inactive if this parameter is set to FALSE. If `datatooltip=TRUE`, the user can select with the mouse a solution in order to have the following information: • 1) value of `k` which has been selected • 2) value of `c` which has been selected • 3) values of the information criterion • 4) frequency distribution of the associated classification.

If datatooltip is a list it may contain the following fields:

1. DisplayStyle determines how the data cursor displays. Possible values are 'datatip' and 'window' (default). 'datatip' displays data cursor information in a small yellow text box attached to a black square marker at a data point you interactively select. 'window' displays data cursor information for the data point you interactively select in a floating window within the figure.
2. SnapToDataVertex: specifies whether the data cursor snaps to the nearest data value or is located at the actual pointer position. Possible values are SnapToDataVertex='on' (default) and SnapToDataVertex='off'.

databrush — Interactive mouse brushing. If databrush is missing or empty (default), no brushing is done. The activation of this option (databrush is TRUE or a list) enables the user to select a set of values of IC in the current plot and to see thecorresponding classification highlighted in the scatterplot matrix. If the scatterplot matrix does not exist it is automatically created. Note that the window style of the other figures is set equal to that which contains the IC plot. In other words, if the IC plot is docked all the other figures will be docked too.

If databrush=TRUE the default selection tool is a rectangular brush and it is possible to brush only once (that is persist=").

If databrush=list(...), it is possible to use all optional arguments of the MATLAB function selectdataFS() and the following optional arguments:

- persist: Persist is an empty value or a character containing 'on' or 'off'. The default value is persist="", that is brushing is allowed only once. If persist="on" or persis="off" brushing can be done as many time as the user requires. If persist='on' then the unit(s) currently brushed are added to those previously brushed. It is possible, every time a new brushing is done, to use a different color for the brushed units. If persist='off' every time a new brush is performed units previously brushed are removed.
- label: add labels of brushed units in the monitoring plot.
- dispopt: controls how to fill the diagonals in the scatterplot matrix of the brushed solutions. Set dispopt="hist" (default) to plot histograms, or dispopt="box" to plot boxplots.

nameY — Add variable labels in plot. A vector of strings of length p containing the labels of the variables in the dataset. If it is empty (default) the sequence X1, ..., Xp will be created automatically

trace — Whether to print intermediate results. Default is trace=FALSE.

... — potential further arguments passed to lower level functions.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Cerioli, A., Garcia-Escudero, L.A., Mayo-Iscar, A. and Riani M. (2017). Finding the Number of Groups in Model-Based Clustering via Constrained Likelihoods, *Journal of Computational and Graphical Statistics*, pp. 404-416, https://doi.org/10.1080/10618600.2017.1390469.

Hubert L. and Arabie P. (1985), Comparing Partitions, *Journal of Classification*, Vol. 2, pp. 193-218.

See Also

tclustIC, tclustfsda

Examples

```
## Not run:
data(geyser2)
out <- tclustIC(geyser2, whichIC="MIXMIX", plot=FALSE, alpha=0.1)

tclustICplot(out, whichIC="MIXMIX")


## End(Not run)
```

---

tclustICsol *Extracts a set of best relevant solutions obtained by tclustIC*

Description

The function `tclustICsol()` takes as input an object of class `tclustic.object`, the output of function `tclustIC` (that is a series of matrices which contain the values of the information criteria BIC/ICL/CLA for different values of k and c) and extracts the first best solutions. Two solutions are considered equivalent if the value of the adjusted Rand index (or the adjusted Fowlkes and Mallows index) is above a certain threshold. For each tentative solution the program checks the adjacent values of c for which the solution is stable. A matrix with adjusted Rand indexes is given for the extracted solutions.

Usage

```
tclustICsol(
  out,
  NumberOfBestSolutions = 5,
  ThreshRandIndex = 0.7,
  whichIC = c("ALL", "CLACLA", "MIXMIX", "MIXCLA"),
  Rand = TRUE,
  msg = TRUE,
  plot = FALSE,
  trace = FALSE,
  ...
)
```

Arguments

out	An S3 object of class tclustic.object (output of tclustIC) containing the values of the information criteria BIC (MIXMIX), ICL (MIXCLA) or CLA (CLACLA), for different values of k (number of groups) and different values of c (restriction factor), for a prespecified level of trimming.
NumberOfBestSolutions	Number of best solutions to extract from BIC/ICL matrix. The default value of NumberOfBestSolutions is 5
ThreshRandIndex	Threshold to identify spurious solutions - the threshold of the adjusted Rand index to use to consider two solutions as equivalent. The default value of ThreshRandIndex is 0.7
whichIC	Specifies the information criterion to use to extract best solutions. Possible values for whichIC are: • CLACLA = in this case best solutions are referred to the classification likelihood. • MIXMIX = in this case in this case best solutions are referred to the mixture likelihood (BIC). • MIXCLA = in this case in this case best solutions are referred to ICL. • ALL = in this case best solutions both three solutions using classification and mixture likelihood are produced. In the output class out all the three matrices MIXMIXbs, CLACLAbs and MIXCLAbs are given. The default value is whichIC="ALL".
Rand	Index to use to compare partitions. If Rand=TRUE (default) the adjusted Rand index is used, else the adjusted Fowlkes and Mallows index is used.
msg	It controls whether to display or not messages (from MATLAB) on the screen. If msg=TRUE (default) messages about the progression of the search are displayed on the screen otherwise only error messages will be displayed.
plot	If plot=TRUE, the best solutions which have been found are shown on the screen.
trace	Whether to print intermediate results. Default is trace=FALSE.
...	potential further arguments passed to lower level functions.

Value

An S3 object of class tclusticsol.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Cerioli, A., Garcia-Escudero, L.A., Mayo-Iscar, A. and Riani M. (2017). Finding the Number of Groups in Model-Based Clustering via Constrained Likelihoods, *Journal of Computational and Graphical Statistics*, pp. 404-416, https://doi.org/10.1080/10618600.2017.1390469.

Hubert L. and Arabie P. (1985), Comparing Partitions, *Journal of Classification*, Vol. 2, pp. 193-218.

See Also

tclustIC, tclustfsda, carbikeplot

Examples

```
## Not run:
data(geyser2)
out <- tclustIC(geyser2, whichIC="MIXMIX", plot=FALSE, alpha=0.1)

## Plot first two best solutions using as Information criterion MIXMIX
print("Best solutions using MIXMIX")
outMIXMIX <- tclustICsol(out, whichIC="MIXMIX", plot=TRUE, NumberOfBestSolutions=2)

print(outMIXMIX$MIXMIXbs)

## End(Not run)
```

---

tclusticsol.object *Objects returned by the function* tclustICsol

---

Description

An object of class tclusticsol.object holds information about the result of a call to tclustICsol.

Value

The functions print() and summary() are used to obtain and print a summary of the results. An object of class tclusticsol is a list containing at least the following components:

call	the matched call
kk	a vector containing the values of k (number of components) which have been considered. This vector is identical to the optional argument kk (default is kk=1:5.
cc	a vector containing the values of c (values of the restriction factor) which have been considered. This vector is identical to the optional argument cc (defalt is cc=c(1, 2, 4, 8, 16, 32, 64, 128).
alpha	trimming level
whichIC	Information criteria used
MIXMIXbs	a matrix of lists of size NumberOfBestSolutions-times-5 which contains the details of the best solutions for MIXMIX (BIC). Each row refers to a solution. The information which is stored in the columns is as follows. • 1st col = value of k for which solution takes place • 2nd col = value of c for which solution takes place; • 3rd col = a vector of length d which contains the values of c for which the solution is uniformly better.

- 4th col = a vector of length d + r which contains the values of c for which the solution is considered stable (i.e. for which the value of the adjusted Rand index (or the adjusted Fowlkes and Mallows index) does not go below the threshold defined in input option ThreshRandIndex).
- 5th col = string which contains 'true' or 'spurious'. The solution is labelled spurious if the value of the adjusted Rand index with the previous solutions is greater than ThreshRandIndex.

Remark: the field MIXMIXbs is present only if whichIC=ALL or whichIC="MIXMIX".

MIXMIXbsari — a matrix of adjusted Rand indexes (or Fowlkes and Mallows indexes) associated with the best solutions for MIXMIX. A matrix of size NumberOfBestSolutions-times-NumberOfBestSo whose i,j-th entry contains the adjusted Rand index between classification produced by solution i and solution j, i,j=1,2, ...,NumberOfBestSolutions.

Remark: the field MIXMIXbsari is present only if whichIC=ALL or whichIC="MIXMIX".

ARIMIX — a matrix of adjusted Rand indexes between two consecutive value of c. Matrix of size k-by-length(cc)-1. The first column contains the ARI indexes between cc[2] and cc[1] given k. The second column contains the the ARI indexes between cc[3] and cc[2] given k.

Remark: the field ARIMIX is present only if whichIC=ALL or whichIC="MIXMIX" or whichIC="MIXCLA".

MIXCLAbs — has the same structure as MIXMIXbs but referres to MIXCLA.

Remark: the field MIXCLAbs is present only if whichIC=ALL or whichIC="MIXCLA".

MIXCLAbsari — has the same structure as MIXMIXbsari but referres to MIXCLA.

Remark: the field MIXMIXbsari is present only if whichIC=ALL or whichIC="MIXCLA".

CLACLAbs — has the same structure as MIXMIXbs but referres to CLACLA.

Remark: the field CLACLAbs is present only if whichIC=ALL or whichIC="CLACLA".

CLACLAbsari — has the same structure as MIXMIXbsari but referres to CLACLA.

Remark: the field CLACLAbsari is present only if whichIC=ALL or whichIC="CLACLA".

ARICLA — a matrix of adjusted Rand indexes between two consecutive value of c. Matrix of size k-by-length(cc)-1. The first column contains the ARI indexes between cc[2] and cc[1] given k. The second column contains the the ARI indexes between cc[3] and cc[2] given k.

Remark: the field ARICLA is present only if whichIC=ALL or whichIC="CLACLA".

See Also

tclustICsol, carbikeplot

Examples

```
## Not run:
data(hbk, package="robustbase")
(out <- tclustIC(hbk[, 1:3]))

 ## Plot first two best solutions using as Information criterion MIXMIX
 print("Best solutions using MIXMIX")
 outMIXMIX <- tclustICsol(out, whichIC="MIXMIX", plot=TRUE, NumberOfBestSolutions=2)
```

```
  class(outMIXMIX)
  summary(outMIXMIX)
  print(outMIXMIX$MIXMIXbs)

## End(Not run)
```

`tclustreg` *Computes robust linear grouping analysis*

Description

Performs robust linear grouping analysis.

Usage

```
tclustreg(
  y,
  x,
  k,
  alphaLik = 0,
  alphaX = 1,
  restrfactor = 12,
  intercept = TRUE,
  plot = FALSE,
  nsamp,
  refsteps = 10,
  reftol = 1e-13,
  equalweights = FALSE,
  mixt = 0,
  wtrim = 0,
  we,
  msg = TRUE,
  RandNumbForNini,
  trace = FALSE,
  ...
)
```

Arguments

y	Response variable. A vector with n elements that contains the response variable.
x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
k	Number of groups.

`alphaLik` Trimming level, a scalar between 0 and 0.5 or an integer specifying the number of observations which have to be trimmed. If `alphaLik=0`, there is no trimming. More in detail, if `0 < alphaLik < 1` clustering is based on `h = floor(n * (1 - alphaLik))` observations. If `alphaLik` is an integer greater than 1 clustering is based on `h = n - floor(alphaLik)`. The likelihood contributions are sorted and the units associated with the smallest `n - h` contributions are trimmed.

`alphaX` Second-level trimming or constrained weighted model for `x`.

- If `alphaX=0` there is no second-level trimming.
- If `alphaX` is in the interval [0, 0.5] it indicates the fixed proportion of units subject to second level trimming. In this case alphaX is usually smaller than alphaLik. For further details see Garcia-Escudero et. al. (2010).
- If `alphaX` is in the interval (0.5 1), it indicates a Bonferronized confidence level to be used to identify the units subject to second level trimming. In this case the fixed a priori, but is determined adaptively. For further details see Torti et al. (2018).
- If `alphaX=1`, constrained weighted model for `X` is assumed (Gershenfeld, 1997). The CWM estimator is able to take into account different distributions for the explanatory variables across groups, so overcoming an intrinsic limitation of mixtures of regression, because they are implicitly assumed equally distributed. Note that if `alphaX=1` it is also possible to apply using restrfactor(2) the constraints on the cov matrices of the explanatory variables. For further details about CWM see Garcia-Escudero et al. (2017) or Torti et al. (2018).

`restrfactor` Restriction factor for regression residuals and covariance matrices of the explanatory variables. Scalar or vector with two elements. If `restrfactor` is a scalar it controls the differences among group scatters of the residuals. The value 1 is the strongest restriction. If `restrfactor` is a vector with two elements the first element controls the differences among group scatters of the residuals and the second the differences among covariance matrices of the explanatory variables. Note that `restrfactor[2]` is used just if `alphaX=1`, that is if constrained weighted model for `x` is assumed.

`intercept` wheather to use constant term (default is `intercept=TRUE`

`plot` If `plot=FALSE` (default) or `plot=0` no plot is produced. If `plot=TRUE` a plot with the final allocation is shown (using the spmplot function). If `X` is 2-dimensional, the lines associated to the groups are shown too.

`nsamp` If a scalar, it contains the number of subsamples which will be extracted. If `nsamp = 0` all subsets will be extracted. Remark - if the number of all possible subset is greater than 300 the default is to extract all subsets, otherwise just 300. If `nsamp` is a matrix it contains in the rows the indexes of the subsets which have to be extracted. `nsamp` in this case can be conveniently generated by function `subsets()`. `nsamp` must have `k * p` columns. The first `p` columns are used to estimate the regression coefficient of group 1, ..., the last `p` columns are used to estimate the regression coefficient of group `k`.

`refsteps` Number of refining iterations in each subsample. Default is `refsteps=10`. `refsteps = 0` means "raw-subsampling" without iterations.

`reftol` Tolerance of the refining steps. The default value is 1e-14

`equalweights` A logical specifying wheather cluster weights in the concentration and assignment steps shall be considered. If `equalweights=TRUE` we are (ideally) assuming equally sized groups, else if `equalweights = false` (default) we allow for different group weights. Please, check in the given references which functions are maximized in both cases.

`mixt` Specifies whether mixture modelling or crisp assignment approach to model based clustering must be used. In the case of mixture modelling parameter mixt also controls which is the criterion to find the untrimmed units in each step of the maximization. If `mixt>=1` mixture modelling is assumed else crisp assignment. The default value is `mixt=0`, i.e. crisp assignment. Please see for details the provided references. The parameter `mixt` also controls the criterion to select the units to trim. If `mixt = 2` the `h` units are those which give the largest contribution to the likelihood, else if `mixt=1` the criterion to select the `h` units is exactly the same as the one which is used in crisp assignment.

`wtrim` How to use the application weights of the observations - can be a flag taking values in c(0, 1, 2, 3, 4) or a list:

- If `wtrim==0` (no weights), the algorithm reduces to the standard `tclustreg` algorithm.
- If `wtrim==1`, trimming is done by weighting the observations using values specified in vector `we`. In this case, vector `we` must be supplied by the user.
- If `wtrim==2`, trimming is again done by weighting the observations using values specified in vector `we`. In this case, vector `we` is computed from the data as a function of the density estimate pdfe. Specifically, the weight of each observation is the probability of retaining the observation, computed as
$$pretain_{ig} = 1 - pdfe_{ig}/max_{ig}(pdfe_{ig})$$
- If `wtrim==3`, trimming is again done by weighting the observations using values specified in vector `we`. In this case, each element wei of vector `we` is a Bernoulli random variable with probability of success $pdfe_{ig}$. In the clustering framework this is done under the constraint that no group is empty.
- If `wtrim==4`, trimming is done with the tandem approach of Cerioli and Perrotta (2014).

If `wtrim` is a list it can have the following components:

- `wtype_beta` the weight for the beta estimation. It can be one of `c(0, 1, 2, 3)`, as above
- `wtype_obj` the weight for the objective function. It can be:
  - `wtype_obj=0` no weights in the objective function
  - `wtype_obj="Z"` Bernoulli random variable with probability of success $pdfe_{ig}$
  - `wtype_obj="w"` a function of the density estimate pdfe
  - `wtype_obj="Zw"` the product of the two above
  - `wtype_obj="user"` user weights `we`.

`we` Weights. A vector of size `n-by-1` containing application-specific weights Default is a vector of ones.

msg — Controls whether to display or not messages on the screen If msg==TRUE (default) messages are displayed on the screen. If msg=2, detailed messages are displayed, for example the information at iteration level.

RandNumbForNini — pre-extracted random numbers to initialize proportions. Matrix of size k-by-nrow(nsamp) containing the random numbers which are used to initialize the proportions of the groups. This option is effective only if nsamp is a matrix which contains pre-extracted subsamples. The purpose of this option is to enable the user to replicate the results when the function tclustreg() is called using a parfor instruction (as it happens for example in routine IC, where tclustreg() is called through a parfor for different values of the restriction factor). The default is that RandNumbForNini is empty - then uniform random numbers are used.

trace — Whether to print intermediate results. Default is trace=FALSE.

... — potential further arguments passed to lower level functions.

Value

An S3 object of class tclustreg.object

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Mayo-Iscar A. (2016). The joint role of trimming and constraints in robust estimation for mixtures of gaussian factor analyzers, Computational Statistics and Data Analysis", Vol. 99, pp. 131-147.

Garcia-Escudero, L.A., Gordaliza, A., Greselin, F., Ingrassia, S. and Mayo-Iscar, A. (2017), Robust estimation of mixtures of regressions with random covariates, via trimming and constraints, Statistics and Computing, Vol. 27, pp. 377-402.

Garcia-Escudero, L.A., Gordaliza A., Mayo-Iscar A., and San Martin R. (2010). Robust clusterwise linear regression through trimming, Computational Statistics and Data Analysis, Vol. 54, pp.3057-3069.

Cerioli, A. and Perrotta, D. (2014). Robust Clustering Around Regression Lines with High Density Regions. Advances in Data Analysis and Classification, Vol. 8, pp. 5-26.

Torti F., Perrotta D., Riani, M. and Cerioli A. (2019). Assessing Robust Methodologies for Clustering Linear Regression Data, Advances in Data Analysis and Classification, Vol. 13, pp 227-257.

Examples

```
## Not run:
## The X data have been introduced by Gordaliza, Garcia-Escudero & Mayo-Iscar (2013).
## The dataset presents two parallel components without contamination.

data(X)
y1 = X[, ncol(X)]
X1 = X[,-ncol(X), drop=FALSE]
```

```
(out <- tclustreg(y1, X1, k=2, alphaLik=0.05, alphaX=0.01, restrfactor=5, plot=TRUE, trace=TRUE))

(out <- tclustreg(y1, X1, k=2, alphaLik=0.05, alphaX=0.01, restrfactor=2,
        mixt=2, plot=TRUE, trace=TRUE))

##  Examples with fishery data

data(fishery)
X <- fishery

## some jittering is necessary because duplicated units are not treated:
## this needs to be addressed
X <- X + 10^(-8) * abs(matrix(rnorm(nrow(X)*ncol(X)), ncol=2))

y1 <- X[, ncol(X)]
X1 <- X[, -ncol(X), drop=FALSE]

(out <- tclustreg(y1, X1, k=3, restrfact=50, alphaLik=0.04, alphaX=0.01, trace=TRUE))
## Example 2:

## Define some arbitrary weight for the units
    we <- sqrt(X1)/sum(sqrt(X1))

##  tclustreg required parameters
    k <- 2; restrfact <- 50; alpha1 <- 0.04; alpha2 <- 0.01

##  Now tclust is run on each combination of mixt and wtrim options

    cat("\nmixt=0; wtrim=0",
        "\nStandard tclustreg, with classification likelihood and without thinning\n")
    (out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
            mixt=0, wtrim=0, trace=TRUE))

    cat("\nmixt=2; wtrim=0",
        "\nMixture likelihood, no thinning\n")
    (out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
            mixt=2, wtrim=0, trace=TRUE))

    cat("\nmixt=0; wtrim=1",
        "\nClassification likelihood, thinning based on user weights\n")
    (out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
            mixt=0, we=we, wtrim=1, trace=TRUE))

    cat("\nmixt=2; wtrim=1",
        "\nMixture likelihood, thinning based on user weights\n")
    (out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
            mixt=2, we=we, wtrim=1, trace=TRUE))

    cat("\nmixt=0; wtrim=2",
        "\nClassification likelihood, thinning based on retention probabilities\n")
    (out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
            mixt=0, wtrim=2, trace=TRUE))
```

```
cat("\nmixt=2; wtrim=2",
    "\nMixture likelihood, thinning based on retention probabilities\n")
(out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
        mixt=2, wtrim=2, trace=TRUE))

cat("\nmixt=0; wtrim=3",
    "\nClassification likelihood, thinning based on bernoulli weights\n")
(out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
        mixt=0, wtrim=3, trace=TRUE))

cat("\nmixt=2; wtrim=3",
    "\nMixture likelihood, thinning based on bernoulli weights\n")
(out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
        mixt=2, wtrim=3, trace=TRUE))

cat("\nmixt=0; wtrim=4",
    "\nClassification likelihood, tandem thinning based on bernoulli weights\n")
(out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
        mixt=0, wtrim=4, trace=TRUE))

cat("\nmixt=2; wtrim=4",
    "\nMixture likelihood, tandem thinning based on bernoulli weights\n")
(out <- tclustreg(y1, X1, k=k, restrfact=restrfact, alphaLik=alpha1, alphaX=alpha2,
        mixt=2, wtrim=4, trace=TRUE))


## End(Not run)
```

`tclustreg.object` *Objects returned by the function* `tclustreg`

Description

An object of class `tclustreg.object` holds information about the result of a call to `tclustreg`.

Value

The functions `print()` and `summary()` are used to obtain and print a summary of the results. An object of class `tclustreg` is a list containing at least the following components:

`call`	the matched call
`bopt`	a `(p+1)`-times-`k` matrix containing the regression parameters.
`sigma2opt`	a `k`-vector containing the estimated group variances.
`sigma2opt_corr`	a `k`-vector containing the estimated group variances corrected with asymptotic consistency factor.
`muXopt`	a `k-by-p` matrix containing cluster centroid locations. Robust estimate of final centroids of the groups. This output is present only if `alphaX=1`.

`sigmaXopt` a `p-by-p-by-k` array containing estimated constrained covariance covariance matrices of the explanatory variables for the `k` groups. This output is present only if `alphaX=1`.

`cstepopt` the concentration step where the objective function was the largest. This is useful when the objective function is not monotone (e.g. with second level trimming or with thinning).

`subsetopt` the subset id where the objective function was the largest.

`idx` an `n-by-1` vector containing the assignment of each unit to each of the k groups. Cluster names are integer numbers from -2 to `k` where `-1` indicates first level trimmed units and `-2` - second level trimmed units.

`size` a matrix of size `k-by-3`:

- 1st col sequence from -2 to `k`
- 2nd col number of observations in each cluster
- 3rd col percentage of observations in each cluster

Remark: 0 denotes thinned units (if the weights to find thinned units are 0 or 1, -1 indicates first level trimmed units and -2 indicates second level trimmed units).

`postprobopt` `n-times-k` matrix containing the final posterior probabilities. `postprobopt[i,j]` contains the posterior probabilitiy of unit `i` from component (cluster) `j`. For the trimmed units posterior probabilities are 0. This output is always produced (independently of the value of `mixt`).

`MIXMIX` BIC which uses parameters estimated using the mixture loglikelihood and the maximized mixture likelihood as goodness of fit measure. This output is present only if `mixt > 0`.

`MIXCLA` BIC which uses the classification likelihood based on parameters estimated using the mixture likelihood (In some books this quantity is called ICL). This output is present only if `mixt > 0`.

`CLACLA` BIC which uses the classification likelihood based on parameters estimated using the classification likelihood. This output is present only if `mixt > 0`.

`obj` the value of the objective function.

`fullsol` a vector of size `nsamp` which contains the value of the objective function (maximized log likelihood) at the end of the iterative process for each extracted subsample. Note that `max(fullsol)` is equal to `obj`.

`NlogL` -2 log of the classification likelihood. In the presence of full convergence `-NlogL/2` is equal `obj`.

`NlogLmixt` -2 log of the mixture likelihood. In the presence of full convergence `-NlogLmixt/2` is equal to `obj`. If `mixt=0` then `NlogLmixt` is empty.

See Also

`tclustreg`

Examples

```
## Not run:

## The X data have been introduced by Gordaliza, Garcia-Escudero & Mayo-Iscar (2013).
## The dataset presents two parallel components without contamination.

data(X)
y1 = X[, ncol(X)]
X1 = X[,-ncol(X), drop=FALSE]

out <- tclustreg(y1, X1, k=2, alphaLik=0.05, alphaX=0.01, restrfactor=5, trace=TRUE)
class(out)
str(out)

## End(Not run)
```

`tclustregIC` *Computes* `tclustreg` *for different number of groups* `k` *and restriction factors* `c`.

Description

The last two letters stand for 'Information Criterion'. This function computes the values of BIC (MIXMIX), ICL (MIXCLA) or CLA (CLACLA), for different values of `k` (number of groups) and different values of `c` (restriction factor for the variances of the residuals), for a prespecified level of trimming. In order to minimize randomness, given `k`, the same subsets are used for each value of `c`.

Usage

```
tclustregIC(
  y,
  x,
  alphaLik = 0,
  alphaX = 1,
  intercept = TRUE,
  whichIC = c("ALL", "MIXMIX", "MIXCLA", "CLACLA"),
  kk = 1:5,
  cc = c(1, 2, 4, 8, 16, 32, 64, 128),
  ccSigmaX = 12,
  plot = FALSE,
  nsamp,
  refsteps = 10,
  reftol = 1e-13,
  equalweights = FALSE,
  we,
  msg = TRUE,
  nocheck = FALSE,
  RandNumbForNini,
```

```
  startv1 = 1,
  UnitsSameGroup,
  commonslope = FALSE,
  Ysave = TRUE,
  trace = FALSE,
  ...
)
```

Arguments

y	Response variable. A vector with n elements that contains the response variable.
x	An n x p data matrix (n observations and p variables). Rows of x represent observations, and columns represent variables. Missing values (NA's) and infinite values (Inf's) are allowed, since observations (rows) with missing or infinite values will automatically be excluded from the computations.
alphaLik	Trimming level, a number between 0 and 0.5 or an integer number specifying the number of observations which have to be trimmed. If `alphaLik=0`, there is no trimming. More in detail, if `0 < alphaLik < 1` clustering is based on `h = floor(n * (1 - alphaLik))` observations. If `alphaLik` is an integer greater than 1 clustering is based on `h = n - floor(alphaLik)`. The likelihood contributions are sorted and the units associated with the smallest `n - h` contributions are trimmed.
alphaX	Second-level trimming or constrained weighted model for `x`. • If `alphaX=0` there is no second-level trimming. • If `alphaX` is in the interval [0, 0.5] it indicates the fixed proportion of units subject to second level trimming. In this case alphaX is usually smaller than alphaLik. For further details see Garcia-Escudero et. al. (2010). • If `alphaX` is in the interval [0.5, 1], it indicates a Bonferronized confidence level to be used to identify the units subject to second level trimming. In this case the fixed a priori, but is determined adaptively. For further details see Torti et al. (2018). • If `alphaX=1`, constrained weighted model for `X` is assumed (Gershenfeld, 1997). The CWM estimator is able to take into account different distributions for the explanatory variables across groups, so overcoming an intrinsic limitation of mixtures of regression, because they are implicitly assumed equally distributed. Note that if `alphaX=1` it is also possible to apply using restrfactor(2) the constraints on the cov matrices of the explanatory variables. For further details about CWM see Garcia-Escudero et al. (2017) or Torti et al. (2018).
intercept	wheather to use constant term (default is `intercept=TRUE`
whichIC	A character value which specifies which information criteria must be computed for each k (number of groups) and each value of the restriction factor c. Possible values for `whichIC` are: • "MIXMIX": a mixture model is fitted and for computing the information criterion the mixture likelihood is used. This option corresponds to the

use of the Bayesian Information criterion (BIC). In output just the matrix MIXMIX is given.

- "MIXCLA": a mixture model is fitted but to compute the information criterion the classification likelihood is used. This option corresponds to the use of the Integrated Complete Likelihood (ICL). In the output just the matrix MIXCLA is given.
- "CLACLA": everything is based on the classification likelihood. This information criterion will be called CLA. In the output just the matrix CLACLA is given.
- "ALL": both classification and mixture likelihood are used. In this case all three information criteria CLA, ICL and BIC are computed. In the output all three matrices MIXMIX, MIXCLA and CLACLA are given.

kk — an integer vector specifying the number of mixture components (clusters) for which the information criteria are be calculated. By default kk=1:5.

cc — a vector specifying the values of the restriction factor which have to be considered for the variances of the residuals of the regression lines. By default cc=c(1, 2, 4, 8, 16, 32, 64, 128).

ccSigmaX — A number specifying the value of the restriction factor which has to be considered for the covariance matrices of the explanatory variables. The default value is ccsigmaX=12. Note that this option is used only if alphaX=1, that is if constrained weighted model (CWM) for x is assumed.

plot — If plot=FALSE (default) or plot=0 no plot is produced. If plot=TRUE a plot with the final allocation is shown (using the spmplot function). If X is 2-dimensional, the lines associated to the groups are shown too.

nsamp — If a scalar, it contains the number of subsamples which will be extracted. If nsamp = 0 all subsets will be extracted. Remark - if the number of all possible subset is greater than 300 the default is to extract all subsets, otherwise just 300. If nsamp is a matrix it contains in the rows the indexes of the subsets which have to be extracted. nsamp in this case can be conveniently generated by function subsets(). nsamp must have k * p columns. The first p columns are used to estimate the regression coefficient of group 1, ..., the last p columns are used to estimate the regression coefficient of group k.

refsteps — Number of refining iterations in each subsample. Default is refsteps=10. refsteps = 0 means "raw-subsampling" without iterations.

reftol — Tolerance of the refining steps. The default value is 1e-14

equalweights — A logical specifying wheather cluster weights in the concentration and assignment steps shall be considered. If equalweights=TRUE we are (ideally) assuming equally sized groups, else if equalweights = false (default) we allow for different group weights. Please, check in the given references which functions are maximized in both cases.

we — Weights. A vector of size n-by-1 containing application-specific weights Default is a vector of ones.

msg — Controls whether to display or not messages on the screen If msg==TRUE (default) messages are displayed on the screen. If msg=2, detailed messages are displayed, for example the information at iteration level.

nocheck — Check input arguments. If nocheck=TRUE no check is performed on matrix X. The default is nocheck=FALSE.

RandNumbForNini — pre-extracted random numbers to initialize proportions. Matrix of size k-by-nrow(nsamp) containing the random numbers which are used to initialize the proportions of the groups. This option is effective only if nsamp is a matrix which contains pre-extracted subsamples. The purpose of this option is to enable the user to replicate the results when the function tclustreg() is called using a parfor instruction (as it happens for example in routine IC, where tclustreg() is called through a parfor for different values of the restriction factor). The default is that RandNumbForNini is empty - then uniform random numbers are used.

startv1 — How to initialize centroids and covariance matrices. Scalar. If startv1=1 then initial centroids and covariance matrices are based on (p+1) observations randomly chosen, else each centroid is initialized taking a random row of input data matrix and covariance matrices are initialized with identity matrices. The default value isstartv1=1.

Remark 1: in order to start with a routine which is in the required parameter space, eigenvalue restrictions are immediately applied.

Remark 2 - option startv1 is used only if nsamp is a scalar for more details see the help associated with nsamp).

UnitsSameGroup — List of the units which must (whenever possible) have a particular label. For example UnitsSameGroup=c(20, 26), means that group which contains unit 20 is always labelled with number 1. Similarly, the group which contains unit 26 is always labelled with number 2, (unless it is found that unit 26 already belongs to group 1). In general, group which contains unit UnitsSameGroup(r) where r=2, ...length(kk)-1 is labelled with number r (unless it is found that unit UnitsSameGroup(r) has already been assigned to groups 1, 2, ..., r-1. The default value of UnitsSameGroup is an empty list, that is consistent labels are not imposed.

commonslope — wheather to impose a constraint of common slope on the regression coefficients. If commonslope=TRUE, the groups are forced to have the same regression coefficients (apart from the intercepts). The default value of commonslope is commonslope=FALSE.

Ysave — weather to save on output the unput response variable y and matrix of predictors x.

trace — Whether to print intermediate results. Default is trace=FALSE.

... — potential further arguments passed to lower level functions.

Value

An S3 object of class tclustregic which is basically a list with the following componnts

- call the matched call
- CLACLA A matrix of size 5-times-8 if kk and cc are not specififed else it is a matrix of size length(kk)-times-length(cc) containinig the value of the penalized classification likelihood. This output is present only if whichIC="CLACLA") or whichIC="ALL").

- IDXCLA array of size 5-times-8 if kk and cc are not specififed else it is an array of size length(kk)-times-length(cc). Each element of the array is a list with one element which is a vector of length n containinig the assignment of each unit using the classification model. This output is present only if whichIC="CLACLA") or whichIC="ALL").
- MIXMIX A matrix of size 5-times-8 if kk and cc are not specififed else it is a matrix of size length(kk)-times-length(cc) containinig the value of the penalized mixture likelihood. This output is present only if whichIC="MIXMIX") or whichIC="ALL").
- MIXCLA A matrix of size 5-times-8 if kk and cc are not specififed else it is a matrix of size length(kk)-times-length(cc) containinig the value of the ICL. This output is present only if whichIC="MIXCLA") or whichIC="ALL").
- IDXMIX array of size 5-times-8 if kk and cc are not specififed else it is an array of size length(kk)-times-length(cc). Each element of the array is a list with one element which is a vector of length n containinig the assignment of each unit using the mixture model. This output is present only if whichIC="MIXMIX"), whichIC="MIXCLA") or whichIC="ALL").
- kk a vector containing the values of k (number of components) which have been considered. This vector is identical to the argument kk (default is kk=1:5.
- cc a vector containing the values of c (values of the restriction factor) which have been considered for the variance of the residuals. This vector is identical to the argument cc (defalt is cc=c(1, 2, 4, 8, 16, 32, 64, 128).
- ccSigmaX values of the restriction factor which have been considered for the covariance matrices of the esplnatory variables. This vector is identical the argument ccsigmaX.
- alpha the trimming level which has been used in the likelidood (it stores the values of input alphaLik).
- alphaX second-level trimming or constrained weighted model for X.
- X original data matrix of explanatory variables. Present if Ysave=TRUE.
- y original vector containing the response. Present if Ysave=TRUE.

Author(s)

FSDA team, <valentin.todorov@chello.at>

References

Torti F., Perrotta D., Riani, M. and Cerioli A. (2019). Assessing Robust Methodologies for Clustering Linear Regression Data, Advances in Data Analysis and Classification, Vol. 13, pp 227-257.

Examples

```
## Not run:
## The X data have been introduced by Gordaliza, Garcia-Escudero & Mayo-Iscar (2013).
## The dataset presents two parallel components without contamination.

data(X)
y1 = X[, ncol(X)]
X1 = X[,-ncol(X), drop=FALSE]

(out <- tclustregIC(y1, X1, plot=TRUE))
```

```
 tclustICplot(out, whichIC="MIXMIX")

## End(Not run)
```

---

wool *Wool data.*

Description

The wool data give the number of cycles to failure of a worsted yarn under cycles of repeated loading. The variables are: length of test specimen; amplitude of loading cycle; load

Usage

```
data(wool)
```

Format

A data frame with 27 rows and 4 variables

X *Simulated data X.*

Description

The X dataset has been simulated by Gordaliza, Garcia-Escudero and Mayo-Iscar during the Workshop ADVANCES IN ROBUST DATA ANALYSIS AND CLUSTERING held in Ispra on October 21st-25th 2013. It is a bivariate dataset of 200 observations. It presents two parallel components without contamination.

Usage

```
data(X)
```

Format

A data frame with 200 rows and 2 variables

z1 *z1*

Description

Simulated data to test tclustIC() and tclustICsol(), carbike() functions

Usage

```
data(z1)
```

Format

A data frame with 150 rows and 2 variables. The variables are as follows:

- X1
- X2

References

Maitra, R. and Melnykov, V. (2010), Simulating data to study performance of finite mixture modeling and clustering algorithms, *The Journal of Computational and Graphical Statistics*, Vol. 19, pp. 354-376.

Examples

```
data(z1)
head(z1)
## Not run:
(out <- tclustIC(z1, plots=FALSE, whichIC="CLACLA"))
(outCLACLA <- tclustICsol(out, whichIC="CLACLA", plot=FALSE))
carbikeplot(outCLACLA)

## End(Not run)
```

Index

∗ **datasets**
bank_data, 4
diabetes, 11
emilia2001, 12
fishery, 16
fishery2003, 17
flea, 17
forbes, 18
geyser2, 52
hawkins, 53
hospital, 53
Income1, 54
Income2, 55
loyalty, 60
M5data, 63
multiple_regression, 90
mussels, 91
poison, 99
swissbanknotes, 141
swissheads, 142
wool, 177
X, 177
z1, 178
∗ **multivariate**
fsmeda.object, 21
fsmmmdrs.object, 25
fsmult.object, 30
fsrfan.object, 51
mmmult.object, 82
mmmulteda.object, 83
score.object, 117
scoreYJ.object, 119
scoreYJpn.object, 122
smult.object, 126
smulteda.object, 127
summary.fsdalms, 137
summary.fsdalts, 138
summary.fsr, 140
tclusteda.object, 143
tclustfsda.object, 152
tclustic.object, 158
tclusticsol.object, 163
tclustreg.object, 170
∗ **regression**
fsdalms.object, 19
fsdalts.object, 20
fsr.object, 31
FSR_control, 32
fsrbase, 35
fsreda.object, 38
FSReda_control, 40
fsreg, 41
levfwdplot, 55
LXS_control, 61
mdrplot, 70
mmreg.object, 84
MMreg_control, 85
mmregeda.object, 87
MMregeda_control, 88
resfwdplot, 106
resindexplot, 111
sreg.object, 131
Sreg_control, 132
sregeda.object, 134
Sregeda_control, 135
∗ **robust**
fsdalms.object, 19
fsdalts.object, 20
fsmeda.object, 21
fsmmmdrs.object, 25
fsmult.object, 30
fsr.object, 31
FSR_control, 32
fsrbase, 35
fsreda.object, 38
FSReda_control, 40
fsreg, 41
fsrfan.object, 51

levfwdplot, 55
LXS_control, 61
mdrplot, 70
mmmult.object, 82
mmmulteda.object, 83
mmreg.object, 84
MMreg_control, 85
mmregeda.object, 87
MMregeda_control, 88
resfwdplot, 106
resindexplot, 111
score.object, 117
scoreYJ.object, 119
scoreYJpn.object, 122
smult.object, 126
smulteda.object, 127
sreg.object, 131
Sreg_control, 132
sregeda.object, 134
Sregeda_control, 135
summary.fsdalms, 137
summary.fsdalts, 138
summary.fsr, 140
tclusteda.object, 143
tclustfsda.object, 152
tclustic.object, 158
tclusticsol.object, 163
tclustreg.object, 170

bank_data, 4

carbikeplot, 5, *157*, *163, 164*
corfwdplot, 6
CovMMest, *81*
covplot, 8
CovSest, *124*

diabetes, 11

emilia2001, 12

fanplot, 13
fishery, 16
fishery2003, 17
flea, 17
forbes, 18
formula, *36*, *42*, *46*, *115*
fsdalms.object, 19, *19*, *43*, *112*
fsdalts.object, 20, *20*, *43*, *112*
fsmeda.object, *7*, *9*, 21, *21*, *29*, *65*, *74*, *128*
fsmmmdrs, 22, *25*, *77*
fsmmmdrs.object, *24*, 25, *25*, *77*
fsmult, *7*, *9*, *21*, 26, *30*, *65*, *69*, *74*, *128*
fsmult.object, *29*, 30, *30*, *69*
fsr.object, 31, *31*, *37*, *43*, *112*
FSR_control, 32, *37*, *41*, *63*, *87*, *90*, *112*, *134*, *137*
fsrbase, 35
fsreda.object, *7*, *15*, *37*, 38, *38*, *43*, *56*, *58*, *67*, *103*, *107*, *109*
FSReda_control, *35*, 40, *71*, *107*
fsreg, *7*, *19, 20*, *31*, *38*, 41, *84*, *87*, *103*, *131*, *134*, *137–140*
fsrfan, *13*, 44, *51*
fsrfan.object, *13*, *48*, 51, *51*

geyser2, 52

hawkins, 53
hospital, 53

Income1, 54
Income2, 55
invisible, *138–140*

levfwdplot, 55
list, *19–21*, *25*, *30, 31*, *35*, *38*, *41*, *63*, *82–84*, *87*, *89*, *126, 127*, *131*, *134*, *137*
logical, *36*, *42*, *46*, *115*
loyalty, 60
LXS_control, *35*, *41*, 61, *87*, *90*, *112*, *134*, *137*

M5data, 63
malfwdplot, 64
malindexplot, 69
mdrplot, 70, *105*
mmdplot, 74
mmdrsplot, *23*, 77
mmmult, *7*, *69*, 80, *82, 83*
mmmult.object, *69*, *81*, 82, *82*
mmmulteda.object, *7*, *81*, 83, *83*
mmreg.object, *43*, 84, *84*, *112*
MMreg_control, *35*, *41*, *63*, 85, *87*, *90*, *112*, *134*, *137*
mmregeda.object, *7*, *15*, *43*, *58*, *67*, 87, *87*, *107*, *109*
MMregeda_control, *35*, 88, *107*
model.matrix.default, *36*, *42*, *46*, *115*

multiple_regression, 90
mussels, 91
myrng, 91

na.exclude, *36*, *42*, *46*, *115*
na.fail, *36*, *42*, *46*, *115*
na.omit, *36*, *42*, *46*, *115*
normBoxCox, 92
normYJ, 95
normYJpn, 97

offset, *36*, *42*, *46*, *115*
options, *36*, *42*, *46*, *115*

plot.fsrfan *(*fsrfan*)*, 44
poison, 99
print, *138–140*
print.fsdalms, *138*
print.fsdalms *(*fsreg*)*, 41
print.fsdalts, *139*
print.fsdalts *(*fsreg*)*, 41
print.fsr, *140*
print.fsr *(*fsreg*)*, 41
print.fsreda *(*fsreg*)*, 41
print.mmreg *(*fsreg*)*, 41
print.mmregeda *(*fsreg*)*, 41
print.sreg *(*fsreg*)*, 41
print.sregeda *(*fsreg*)*, 41
print.summary.fsdalms
*(*summary.fsdalms*)*, 137
print.summary.fsdalts
*(*summary.fsdalts*)*, 138
print.summary.fsr *(*summary.fsr*)*, 140
printCoefmat, *137*, *139*, *140*
psifun, 99

regspmplot, 102
resfwdplot, *105*, 106
resindexplot, 111

score, 114, *117*
score.object, *116*, 117, *117*
scoreYJ, 117, *119*
scoreYJ.object, *118*, 119, *119*
scoreYJpn, 120, *122*
scoreYJpn.object, *121*, 122, *122*
smult, *7*, *69*, 123, *126*, *127*
smult.object, *69*, *124*, 126, *126*
smulteda.object, *7*, *124*, 127, *127*

spmplot, *105*, 128
sreg.object, *43*, *112*, 131, *131*
Sreg_control, *35*, *63*, *88*, *90*, *112*, 132
sregeda.object, *7*, *15*, *43*, *58*, *67*, *107*, *109*, 134, *134*
Sregeda_control, *35*, *107*, 135
summary, *137–140*
summary.fsdalms, 137, *138*
summary.fsdalts, 138, *139*
summary.fsr, 140, *140*
swissbanknotes, 141
swissheads, 142

tclusteda.object, 143, *143*, *149*
tclustfsda, *143*, 144, *152*, *154*, *157*, *161*, *163*
tclustfsda.object, *144*, *149*, 152, *152*
tclustIC, 154, *158*, *159*, *161–163*
tclustic.object, *157*, 158, *158*, *159*, *161*, *162*
tclustICplot, *157*, 159
tclustICsol, *5*, *157*, 161, *163*, *164*
tclusticsol.object, *5*, *162*, 163, *163*
tclustreg, 165, *170*, *171*
tclustreg.object, *168*, 170, *170*
tclustregIC, 172

wool, 177

X, 177

z1, 178