Package 'RobStatTM'

April 23, 2019

Version 1.0.0

Date 2019-03-03

Title Robust Statistics: Theory and Methods

Description Companion package for the book: ``Robust Statistics: Theory and Methods, second edition", <http://www.wiley.com/go/maronna/robust>. This package contains code that implements the robust estimators discussed in the recent second edition of the book above, as well as the scripts reproducing all the examples in the book.

Depends R (>= 3.0.2), fit.models

Imports stats, graphics, utils, methods, DEoptimR, pyinit, rrcov, robustbase, shiny, shinyjs, PerformanceAnalytics, DT, ggplot2, gridExtra, robust, xts

Suggests knitr

LazyData yes

License GPL (>= 3)

RoxygenNote 6.1.1

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation yes

Author Matias Salibian-Barrera [cre],
Victor Yohai [aut],
Ricardo Maronna [aut],
Doug Martin [aut],
Gregory Brownson [aut] (ShinyUI),
Kjell Konis [aut],
Kjell Konis [cph] (erfi),
Christophe Croux [ctb] (WBYlogreg, BYlogreg),
Gentiane Haesbroeck [ctb] (WBYlogreg, BYlogreg),
Martin Maechler [cph] (lmrob.fit, lmrob..M..fit, lmrob.S),
Manuel Koller [cph] (lmrob.fit, .vcov.avar1, lmrob.S, lmrob.lar),
Matias Salibian-Barrera [aut]

Maintainer Matias Salibian-Barrera <matias@stat.ubc.ca>

Repository CRAN

Date/Publication 2019-04-23 11:30:06 UTC

R topics documented:

`algae` *Algae data*

Description

Algae data. More details here.

Usage

```
data(algae)
```

Format

An object of class `"data.frame"`.

Source

Source goes here.

References

References go here.

Examples

```
data(algae)
```

`biochem` *Biochem data*

Description

Biochem data. More details here.

Usage

```
data(biochem)
```

Format

An object of class `"data.frame"`.

Source

Source goes here.

References

References go here.

Examples

```
data(biochem)
```

`bisquare` *Tuning parameter the rho loss functions*

Description

This function computes the tuning constant that yields an MM-regression estimator with a desired asymptotic efficiency when computed with a rho function in the corresponding family. The output of this function can be passed to the functions lmrobdet.control, mscale and rho.

Usage

```
bisquare(e)
```

Arguments

`e`	the desired efficiency of the corresponding regression estimator for Gaussian errors

Value

A length-1 vector with the corresponding tuning constant.

Author(s)

Kjell Konis

Examples

```
# Tuning parameters for an 85%-efficient M-estimator at a Gaussian model
bisquare(.85)
```

bus *Bus data*

Description

Bus data. More details here.

Usage

```
data(bus)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(bus)
```

cov.dcml *Approximate covariance matrix of the DCML regression estimator.*

Description

The estimated covariance matrix of the DCML regression estimator. This function is used internally and not meant to be used directly.

Usage

```
cov.dcml(res.LS, res.R, CC, sig.R, t0, p, n, control)
```

Arguments

res.LS	vector of residuals from the least squares fit
res.R	vector of residuals from the robust regression fit
CC	estimated covariance matrix of the robust regression estimator
sig.R	robust estimate of the scale of the residuals
t0	mixing parameter
p, n	the dimensions of the problem, needed for the finite sample correction of the tuning constant of the M-scale
control	a list of control parameters as returned by lmrobdet.control

Value

The covariance matrix estimate.

Author(s)

Victor Yohai, <victoryohai@gmail.com>

---

covClassic *Classical Covariance Estimation*

Description

Compute an estimate of the covariance/correlation matrix and location vector using classical methods.

Usage

```
covClassic(data, corr = FALSE, center = TRUE, distance = TRUE,
  na.action = na.fail, unbiased = TRUE)
```

Arguments

data	a numeric matrix or data frame containing the data.
corr	a logical flag. If corr = TRUE then the estimated correlation matrix is computed.
center	a logical flag or a numeric vector of length p (where p is the number of columns of x) specifying the center. If center = TRUE then the center is estimated. Otherwise the center is taken to be 0.
distance	a logical flag. If distance = TRUE the Mahalanobis distances are computed.
na.action	a function to filter missing data. The default na.fail produces an error if missing values are present. An alternative is na.omit which deletes observations that contain one or more missing values.
unbiased	a logical flag. If TRUE the unbiased estimator is returned (computed with denominator equal to n-1), else the MLE (computed with denominator equal to n) is returned.

Details

Its main intention is to return an object compatible to that produced by covRob, but fit using classical methods.

Value

a list with class "covClassic" containing the following elements:

`call`	an image of the call that produced the object with all the arguments named.
`cov`	a numeric matrix containing the estimate of the covariance/correlation matrix.
`center`	a numeric vector containing the estimate of the location vector.
`dist`	a numeric vector containing the squared Mahalanobis distances. Only present if `distance = TRUE` in the `call`.
`corr`	a logical flag. If `corr = TRUE` then `cov` contains an estimate of the correlation matrix of `x`.

Note

Originally, and in S-PLUS, this function was called `cov`; it has been renamed, as that did mask the function in the standard package **stats**.

Examples

```
data(wine)
round( covClassic(wine)$cov, 2)
```

---

covRob *Robust multivariate location and scatter estimators*

Description

This function computes robust estimators for multivariate location and scatter.

Usage

```
covRob(X, type = "auto", maxit = 50, tol = 1e-04)
```

Arguments

`X`	a data matrix with observations in rows.
`type`	a string indicating which estimator to compute. Valid options are "Rocke" for Rocke's S-estimator, "MM" for an MM-estimator with a SHR rho function, or "auto" (default) which selects "Rocke" if the number of variables is greater than or equal to 10, and "MM" otherwise.
`maxit`	Maximum number of iterations, defaults to 50.
`tol`	Tolerance for convergence, defaults to 1e-4.

Details

This function computes robust estimators for multivariate location and scatter. The default behaviour (`type = "auto"`) computes a "Rocke" estimator (as implemented in `covRobRocke`) if the number of variables is greater than or equal to 10, and an MM-estimator with a SHR rho function (as implemented in `covRobMM`) otherwise.

Value

A list with class "covClassic" with the following components:

`mu`	The location estimate
`V`	The scatter matrix estimate, scaled for consistency at the normal distribution
`center`	The location estimate. Same as `mu` above.
`cov`	The scatter matrix estimate, scaled for consistency at the normal distribution. Same as `V` above.
`dist`	Robust Mahalanobis distances

Author(s)

Ricardo Maronna, `<rmaronna@retina.ar>`

References

http://www.wiley.com/go/maronna/robust

See Also

`covRobRocke`, `covRobMM`

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
tmp <- covRob(X1)
round(tmp$cov[1:10, 1:10], 3)
tmp$mu
```

`covRobMM` *MM robust multivariate location and scatter estimator*

Description

This function computes an MM robust estimator for multivariate location and scatter with the "SHR" loss function.

Usage

```
covRobMM(X, maxit = 50, tolpar = 1e-04)
```

Arguments

`X`	a data matrix with observations in rows.
`maxit`	Maximum number of iterations.
`tolpar`	Tolerance to decide converngence.#'

Details

This function computes an MM robust estimator for multivariate location and scatter with the "SHR" loss function.

Value

A list with class "covRob" containing the following elements

`mu`	The location estimate
`V`	The scatter matrix estimate, scaled for consistency at the normal distribution
`center`	The location estimate. Same as `mu` above.
`cov`	The scatter matrix estimate, scaled for consistency at the normal distribution. Same as `V` above.
`dista`	Robust Mahalanobis distances

Author(s)

Ricardo Maronna, <rmaronna@retina.ar>

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
tmp <- covRobMM(X1)
round(tmp$cov[1:10, 1:10], 3)
tmp$mu
```

---

covRobRocke *Rocke's robust multivariate location and scatter estimator*

Description

This function computes Rocke's robust estimator for multivariate location and scatter.

Usage

```
covRobRocke(X, initial = "K", maxsteps = 5, propmin = 2, qs = 2,
  maxit = 50, tol = 1e-04)
```

Arguments

X	a data matrix with observations in rows.
initial	A character indicating the initial estimator. Valid options are 'K' (default) for the Pena-Prieto 'KSD' estimate, and 'mve' for the Minimum Volume Ellipsoid.
maxsteps	Maximum number of steps for the line search section of the algorithm.
propmin	Regulates the proportion of weights computed from the initial estimator that will be different from zero. The number of observations with initial non-zero weights will be at least p (the number of columns of X) times propmin.
qs	Tuning paramater for Rocke's loss functions.
maxit	Maximum number of iterations.
tol	Tolerance to decide converngence.#'

Details

This function computes Rocke's robust estimator for multivariate location and scatter.

Value

A list with class "covRob" containing the following elements:

mu	The location estimate
V	The scatter matrix estimate, scaled for consistency at the normal distribution
center	The location estimate. Same as mu above.

cov	The scatter matrix estimate, scaled for consistency at the normal distribution. Same as V above.
dista	Robust Mahalanobis distances
w	weights
gamma	Final value of the constant gamma that regulates the efficiency

Author(s)

Ricardo Maronna, <rmaronna@retina.ar>

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
tmp <- covRobRocke(X1)
round(tmp$cov[1:10, 1:10], 3)
tmp$mu
```

DCML *DCML regression estimator*

Description

This function computes the DCML regression estimator. This function is used internally by lmrobdetDCML, and not meant to be used directly.

Usage

```
DCML(x, y, z, z0, control)
```

Arguments

x	design matrix
y	response vector
z	robust fit as returned by MMPY or SMPY
z0	least squares fit as returned by lm.fit
control	a list of control parameters as returned by lmrobdet.control

Value

a list with the following components

coefficients	the vector of regression coefficients
cov	the estimated covariance matrix of the DCML regression estimator
residuals	the vector of regression residuals from the DCML fit
scale	a robust residual (M-)scale estimate
t0	the mixing proportion between the least squares and robust regression estimators

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

---

drop1.lmrobdetMM *RFPE of submodels of an* lmrobdetMM *fit*

Description

This function computes the RFPE for the MM-estimators obtained with lmrobdetMM by recomputing it, successively removing each of a number of specified terms. It is used internally by step.lmrobdetMM and not meant to be used directly.

Usage

```
## S3 method for class 'lmrobdetMM'
drop1(object, scope, scale, keep, ...)
```

Arguments

object	the MM element (of class lmrob) in an object of class lmrobdetMM.
scope	an optional formula giving the terms to be considered for dropping. Typically this argument is omitted, in which case all possible terms are dropped (without breaking hierarchy rules). The scope can also be a character vector of term labels. If the argument is supplied as a formula, any . is interpreted relative to the formula implied by the object argument.
scale	an optional residual scale estimate. If missing the residual scale estimate in object is used.

keep	a character vector of names of components that should be saved for each subset model. Only names from the set "coefficients", "fitted" and "residuals" are allowed. If keep == TRUE, the complete set is saved. The default behavior is not to keep anything.
...	additional parameters to match generic method drop1

Value

An anova object consisting of the term labels, the degrees of freedom, and Robust Final Prediction Errors (RFPE) for each subset model. If keep is missing, the anova object is returned. If keep is present, a list with components "anova" and "keep" is returned. In this case, the "keep" component is a matrix of mode "list", with a column for each subset model, and a row for each component kept.

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

lmrobdetMM

---

fastmve *Minimum Volume Ellipsoid covariance estimator*

Description

This function uses a fast algorithm to compute the Minimum Volume Ellipsoid (MVE) for multivariate location and scatter.

Usage

```
fastmve(x, nsamp = 500)
```

Arguments

x	data matrix (n x p) with cases stored in rows.
nsamp	number of random starts for the iterative algorithm, these are constructed using subsamples of the data.

Details

This function computes the Minimum Volume Ellipsoid (MVE) for multivariate location and scatter, using a fast algorithm related to the fast algorithm for S-regression estimators (see lmrob).

Value

A list with the following components:

center	a vector with the robust multivariate location estimate
cov	a matrix with the robust covariance / scatter matrix estimate
scale	A scalar that equals the median of the mahalanobis distances of the data to the center, multiplied by the determinant of the covariance matrix to the power 1/p
best	Indices of the observations that correspond to the MVE estimator
nsamp	Number of random starts used for the iterative algorithm
nsing	Number of random subsamples (among the nsamp attempted) that failed (resulting in singular initial values)

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
tmp <- fastmve(X1)
round(tmp$cov[1:10, 1:10], 3)
tmp$center
```

---

flour *Flour data*

Description

Flour data. More details here.

Usage

```
data(flour)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(flour)
```

`initPP` *Robust multivariate location and scatter estimators*

Description

This function computes robust multivariate location and scatter estimators using both random and deterministic starting points.

Usage

```
initPP(X, muldirand = 20, muldifix = 10, dirmin = 1000)
```

Arguments

`X`	a data matrix with observations in rows.
`muldirand`	used to determine the number of random directions (candidates), which is `max(p*muldirand, dirmin)`, where `p` is the number of columns in `X`.
`muldifix`	used to determine the number of random directions (candidates), which is `min(n, 2*muldifix*p)`.
`dirmin`	minimum number of random directions

Details

This function computes robust multivariate location and scatter using both Pen~a-Prieto and random candidates.

Value

A list with the following components:

`idx`	A zero/one vector with ones in the positions of the suspected outliers
`disma`	Robust squared Mahalanobis distances
`center`	Robust mean estimate
`cova`	Robust covariance matrix estimate
`t`	Outlyingness of data points

Author(s)

Ricardo Maronna, <rmaronna@retina.ar>, based on original code by D. Pen~a and J. Prieto

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
tmp <- initPP(X1)
round(tmp$cov[1:10, 1:10], 3)
tmp$center
```

INVTR2 *Robust R^2 coefficient of determination*

Description

This function computes a robust version of the R^2 coefficient of determination. It is used internally by lmrobdetMM, and not meant to be used directly.

Usage

```
INVTR2(RR2, family, cc)
```

Arguments

RR2	the proportional difference in loss functions (a naive robust R^2 coefficient).
family	family string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt").
cc	tuning parameters to be computed according to efficiency and / or breakdown considerations. See lmrobdet.control, bisquare, modopt and optimal.

Details

This function computes a robust version of the R^2 coefficient. It is used internally by lmrobdetMM, and not meant to be used directly.

Value

An unbiased version of the robust R^2 coefficient of determination.

Author(s)

Victor Yohai, <victoryohai@gmail.com>

References

http://www.wiley.com/go/maronna/robust

`lmrobdet.control` *Tuning parameters for lmrobdetMM and lmrobdetDCML*

Description

This function sets tuning parameters for the MM estimator implemented in `lmrobdetMM` and the Distance Constrained Maximum Likelihood regression estimators computed by `lmrobdetDCML`.

Usage

```
lmrobdet.control(bb = 0.5, efficiency = 0.99, family = "optimal",
  tuning.psi, tuning.chi, compute.rd = FALSE, corr.b = TRUE,
  split.type = "f", initial = "S", max.it = 100,
  refine.tol = 1e-07, rel.tol = 1e-07, refine.PY = 10,
  solve.tol = 1e-07, trace.lev = 0, psc_keep = 0.5,
  resid_keep_method = "threshold", resid_keep_thresh = 2,
  resid_keep_prop = 0.2, py_maxit = 20, py_eps = 1e-05,
  mscale_maxit = 50, mscale_tol = 1e-06, mscale_rho_fun = "bisquare")
```

Arguments

`bb`	tuning constant (between 0 and 1/2) for the M-scale used to compute the initial S-estimator. It determines the robusness (breakdown point) of the resulting MM-estimator, which is `bb`. Defaults to 0.5.
`efficiency`	desired asymptotic efficiency of the final regression M-estimator. Defaults to 0.85.
`family`	string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt"). Incomplete entries will be matched to the current valid options.
`tuning.psi`	tuning parameters for the regression M-estimator computed with a rho function as specified with argument `family`. If missing, it is computed inside `lmrobdet.control` to match the value of `efficiency` according to the family of rho functions specified in `family`. Appropriate values for `tuning.psi` for a given desired efficiency for Gaussian errors can be constructed using the functions bisquare, modopt and optimal.
`tuning.chi`	tuning constant for the function used to compute the M-scale used for the initial S-estimator. If missing, it is computed inside `lmrobdet.control` to match the value of `bb` according to the family of rho functions specified in `family`.
`compute.rd`	logical value indicating whether robust leverage distances need to be computed.
`corr.b`	logical value indicating whether a finite-sample correction should be applied to the M-scale parameter `bb`.
`split.type`	determines how categorical and continuous variables are split. See `splitFrame`.
`initial`	string specifying the initial value for the M-step of the MM-estimator. Valid options are `'S'`, for an S-estimator and `'MS'` for an M-S estimator which is appropriate when there are categorical explanatory variables in the model.

max.it	maximum number of IRWLS iterations for the MM-estimator
refine.tol	relative covergence tolerance for the S-estimator
rel.tol	relative covergence tolerance for the IRWLS iterations for the MM-estimator
refine.PY	number of refinement steps for the Pen~a-Yohai candidates
solve.tol	(for the S algorithm): relative tolerance for matrix inversion. Hence, this corresponds to solve.default's tol.
trace.lev	positive values (increasingly) provide details on the progress of the MM-algorithm
psc_keep	For pyinit, proportion of observations to remove based on PSCs. The effective proportion of removed observations is adjusted according to the sample size to be prosac*(1-p/n). See pyinit.
resid_keep_method	For pyinit, how to clean the data based on large residuals. If "threshold", all observations with scaled residuals larger than C.res will be removed, if "proportion", observations with the largest prop residuals will be removed. See pyinit.
resid_keep_thresh	See parameter resid_keep_method above. See pyinit.
resid_keep_prop	See parameter resid_keep_method above. See pyinit.
py_maxit	Maximum number of iterations. See pyinit.
py_eps	Relative tolerance for convergence. See pyinit.
mscale_maxit	Maximum number of iterations for the M-scale algorithm. See pyinit and mscale.
mscale_tol	Convergence tolerance for the M-scale algorithm. See mscale and mscale.
mscale_rho_fun	String indicating the loss function used for the M-scale. See pyinit.

Value

A list with the necessary tuning parameters.

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

See Also

pyinit, mscale.

Examples

```
data(coleman, package='robustbase')
m2 <- lmrobdetMM(Y ~ ., data=coleman, control=lmrobdet.control(refine.PY=50))
m2
summary(m2)
```

`lmrobdetDCML` *Robust Distance Constrained Maximum Likelihood estimators for linear regression*

Description

This function computes robust Distance Constrained Maximum Likelihood estimators for linear models.

Usage

```
lmrobdetDCML(formula, data, subset, weights, na.action, model = TRUE,
  x = !control$compute.rd, y = FALSE, singular.ok = TRUE,
  contrasts = NULL, offset = NULL, control = lmrobdet.control())
```

Arguments

`formula`	a symbolic description of the model to be fit.
`data`	an optional data frame, list or environment containing the variables in the model. If not found in `data`, model variables are taken from `environment(formula)`, which usually is the root environment of the current R session.
`subset`	an optional vector specifying a subset of observations to be used.
`weights`	an optional vector of weights to be used in the fitting process.
`na.action`	a function to 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.
`model`	logical value indicating whether to return the model frame
`x`	logical value indicating whether to return the model matrix
`y`	logical value indicating whether to return the vector of responses
`singular.ok`	logical value. If `FALSE` a singular fit produces an error.
`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.
`control`	a list specifying control parameters as returned by the function lmrobdet.control.

Details

This function computes Distance Constrained Maximum Likelihood regression estimators computed using an MM-regression estimator based on Pen~a-Yohai candidates (instead of subsampling ones). This function makes use of the functions `lmrob.fit`, `lmrob..M..fit`, `.vcov.avar1`, `lmrob.S` and `lmrob.lar`, from robustbase, along with utility functions used by these functions, modified so as to include use of the analytic form of the optimal psi and rho functions (for the optimal psi function , see Section 5.8.1 of Maronna, Martin, Yohai and Salibian Barrera, 2019)

Value

A list with the following components:

coefficients	The estimated vector of regression coefficients
scale	The estimated scale of the residuals
residuals	The vector of residuals associated with the robust fit
converged	Logical value indicating whether IRWLS iterations for the MM-estimator have converged
iter	Number of IRWLS iterations for the MM-estimator
rweightsMM	Robustness weights for the MM-estimator
fitted.values	Fitted values associated with the robust fit
rank	Numeric rank of the fitted linear model
cov	The estimated covariance matrix of the regression estimates
df.residual	The residual degrees of freedom
contrasts	(only where relevant) the contrasts used
xlevels	(only where relevant) a record of the levels of the factors used in fitting
call	the matched call
model	if requested, the model frame used
x	if requested, the model matrix used
y	if requested, the response vector used
na.action	(where relevant) information returned by model.frame on the special handling of NAs

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>, based on lmrob

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

Examples

```
data(coleman, package='robustbase')
m1 <- lmrobdetDCML(Y ~ ., data=coleman)
m1
summary(m1)
```

`lmrobdetMM` *Robust linear regression estimators*

Description

This function computes an MM-regression estimators for linear models using deterministic starting points.

Usage

```
lmrobdetMM(formula, data, subset, weights, na.action, model = TRUE,
  x = !control$compute.rd, y = FALSE, singular.ok = TRUE,
  contrasts = NULL, offset = NULL, control = lmrobdet.control())
```

Arguments

`formula`	a symbolic description of the model to be fit.
`data`	an optional data frame, list or environment containing the variables in the model. If not found in `data`, model variables are taken from `environment(formula)`, which usually is the root environment of the current R session.
`subset`	an optional vector specifying a subset of observations to be used.
`weights`	an optional vector of weights to be used in the fitting process.
`na.action`	a function to 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.
`model`	logical value indicating whether to return the model frame
`x`	logical value indicating whether to return the model matrix
`y`	logical value indicating whether to return the vector of responses
`singular.ok`	logical value. If `FALSE` a singular fit produces an error.
`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.
`control`	a list specifying control parameters as returned by the function lmrobdet.control.

Details

This function computes MM-regression estimators computed using Pen~a-Yohai candidates (instead of subsampling ones). This function makes use of the functions `lmrob.fit`, `lmrob..M..fit`, `.vcov.avar1`, `lmrob.S` and `lmrob.lar`, from robustbase, along with utility functions used by these functions, modified so as to include use of the analytic form of the optimal psi and rho functions (for the optimal psi function , see Section 5.8.1 of Maronna, Martin, Yohai and Salibian Barrera, 2019)

Value

A list with the following components:

coefficients	The estimated vector of regression coefficients
scale	The estimated scale of the residuals
residuals	The vector of residuals associated with the robust fit
converged	Logical value indicating whether IRWLS iterations for the MM-estimator have converged
iter	Number of IRWLS iterations for the MM-estimator
rweights	Robustness weights for the MM-estimator
fitted.values	Fitted values associated with the robust fit
rank	Numeric rank of the fitted linear model
cov	The estimated covariance matrix of the regression estimates
df.residual	The residual degrees of freedom
contrasts	(only where relevant) the contrasts used
xlevels	(only where relevant) a record of the levels of the factors used in fitting
call	the matched call
model	if requested, the model frame used
x	if requested, the model matrix used
y	if requested, the response vector used
na.action	(where relevant) information returned by model.frame on the special handling of NAs

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>, based on lmrob from package robustbase

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

Examples

```
data(coleman, package='robustbase')
m2 <- lmrobdetMM(Y ~ ., data=coleman)
m2
summary(m2)
```

lmrobdetMM.RFPE *Robust Final Prediction Error*

Description

This function computes the robust Final Prediction Errors (RFPE) for a robust regression fit using M-estimates. It is used internally by `step.lmrobdetMM` and not meant to be used directly.

Usage

```
lmrobdetMM.RFPE(object, scale = NULL)
```

Arguments

`object`	the `MM` element (of class `lmrob`) in an object of class `lmrobdetMM`.
`scale`	a numeric value specifying the scale estimate used to compute the RFPE. Usually this should be the scale estimate from an encompassing model. If `NULL`, the scale estimate in `object` is used.

Value

the robust final prediction error (numeric).

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

`lmrobdetMM`

lmrobLinTest *Robust likelihood ratio test for linear hypotheses*

Description

This function computes a robust likelihood ratio test for linear hypotheses.

Usage

```
lmrobLinTest(object1, object2)
```

Arguments

object1	an lmrob object with the fit corresponding to the complete model
object2	an lmrob object with the fit corresponding to the model restricted under the null linear hypothesis.

Value

A list with the following components: c("test","chisq.pvalue","f.pvalue","df")

test	The value of the F-statistic
f.pvalue	p-value based on the F distribution
chisq.pvalue	p-value based on the chi-squared distribution
df	degrees of freedom

Author(s)

Victor Yohai, <vyohai@gmail.com>

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(oats)
cont <- lmrobdet.control(bb = 0.5, efficiency = 0.85, family = "bisquare")
oats1M <- lmrobM(response1 ~ variety+block, control=cont, data=oats)
oats1M_var <- lmrobM(response1 ~ block, control=cont, data=oats)
( anov1M_var <- rob.linear.test(oats1M, oats1M_var) )
```

---

lmrobM *Robust estimators for linear regression with fixed designs*

Description

This function computes a robust regression estimator for a linear models with fixed designs.

Usage

```
lmrobM(formula, data, subset, weights, na.action, model = TRUE,
  x = FALSE, y = FALSE, singular.ok = TRUE, contrasts = NULL,
  offset = NULL, control = lmrobM.control())
```

Arguments

formula	a symbolic description of the model to be fit.
data	an optional data frame, list or environment containing the variables in the model. If not found in data, model variables are taken from environment(formula), which usually is the root environment of the current R session.
subset	an optional vector specifying a subset of observations to be used.
weights	an optional vector of weights to be used in the fitting process.
na.action	a function to 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.
model	logical value indicating whether to return the model frame
x	logical value indicating whether to return the model matrix
y	logical value indicating whether to return the vector of responses
singular.ok	logical value. If FALSE a singular fit produces an error.
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.
control	a list specifying control parameters as returned by the function lmrobM.control.

Details

This function computes robust regression estimators for linear models with fixed designs. It computes an L1 estimator, and uses it as a starting point to find a minimum of a re-descending M estimator. The scale is set to a quantile of the absolute residuals from the L1 estimator. This function makes use of the functions lmrob.fit, lmrob..M..fit, .vcov.avar1, lmrob.S and lmrob.lar, from robustbase, along with utility functions used by these functions, modified so as to include use of the analytic form of the optimal psi and rho functions (for the optimal psi function , see Section 5.8.1 of Maronna, Martin, Yohai and Salibian Barrera, 2019)

Value

A list with the following components:

coefficients	The estimated vector of regression coefficients
scale	The estimated scale of the residuals
residuals	The vector of residuals associated with the robust fit
converged	Logical value indicating whether IRWLS iterations for the MM-estimator have converged
iter	Number of IRWLS iterations for the MM-estimator
rweights	Robustness weights for the MM-estimator
fitted.values	Fitted values associated with the robust fit
rank	Numeric rank of the fitted linear model
cov	The estimated covariance matrix of the regression estimates

df.residual	The residual degrees of freedom
contrasts	(only where relevant) the contrasts used
xlevels	(only where relevant) a record of the levels of the factors used in fitting
call	the matched call
model	if requested, the model frame used
x	if requested, the model matrix used
y	if requested, the response vector used
na.action	(where relevant) information returned by model.frame on the special handling of NAs

Author(s)

Victor Yohai, <vyohai@gmail.com>, based on lmrob

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(shock)
cont <- lmrobM.control(bb = 0.5, efficiency = 0.85, family = "bisquare")
shockrob <- lmrobM(time ~ n.shocks, data = shock, control=cont)
shockrob
summary(shockrob)
```

---

lmrobM.control *Tuning parameters for lmrobM*

Description

This function sets tuning parameters for the M estimators of regression implemented in lmrobM.

Usage

```
lmrobM.control(bb = 0.5, efficiency = 0.99, family = "optimal",
  tuning.chi, tuning.psi, max.it = 100, rel.tol = 1e-07,
  mscale_tol = 1e-06, mscale_maxit = 50, trace.lev = 0)
```

Arguments

bb	tuning constant (between 0 and 1/2) for the M-scale used to compute the residual scale estimator. Defaults to 0.5.
efficiency	desired asymptotic efficiency of the final regression M-estimator. Defaults to 0.85.
family	string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt"). Incomplete entries will be matched to the current valid options.
tuning.chi	tuning constant for the function used to compute the M-scale used for the residual scale estimator. If missing, it is computed inside `lmrobdet.control` to match the value of bb according to the family of rho functions specified in `family`.
tuning.psi	tuning parameters for the regression M-estimator computed with a rho function as specified with argument `family`. If missing, it is computed inside `lmrobdet.control` to match the value of `efficiency` according to the family of rho functions specified in `family`. Appropriate values for `tuning.psi` for a given desired efficiency for Gaussian errors can be constructed using the functions bisquare, modopt and optimal.
max.it	maximum number of IRWLS iterations for the M-estimator
rel.tol	relative covergence tolerance for the IRWLS iterations for the M-estimator
mscale_tol	Convergence tolerance for the M-scale algorithm. See `mscale`.
mscale_maxit	Maximum number of iterations for the M-scale algorithm. See `mscale`.
trace.lev	positive values (increasingly) provide details on the progress of the M-algorithm

Value

A list with the necessary tuning parameters.

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

Examples

```
data(coleman, package='robustbase')
m2 <- lmrobM(Y ~ ., data=coleman, control=lmrobM.control())
m2
summary(m2)
```

`locScaleM` *Robust univariate location and scale M-estimators*

Description

This function computes M-estimators for location and scale.

Usage

```
locScaleM(x, psi = "bisquare", eff = 0.9, maxit = 50, tol = 1e-04)
```

Arguments

`x`	a vector of univariate observations
`psi`	a string indicating which score function to use. Valid options are "bisquare", "huber", "optimal" and "modopt".
`eff`	desired asymptotic efficiency. Valid options are 0.85, 0.9 (default) and 0.95 when `psi` = "bisquare" or "huber", and 0.85, 0.9 (default), 0.95 and 0.99 when `psi` = "optimal" or "modopt".
`maxit`	maximum number of iterations allowed.
`tol`	tolerance to decide convergence of the iterative algorithm.

Details

This function computes M-estimators for location and scale.

Value

A list with the following components:

`mu`	The location estimate
`std.mu`	Estimated standard deviation of the location estimator `mu`
`disper`	M-scale/dispersion estimate

Author(s)

Ricardo Maronna, <rmaronna@retina.ar>

References

http://www.wiley.com/go/maronna/robust

Examples

```
set.seed(123)
r <- rnorm(150, sd=1.5)
locScaleM(r)
# 10% of outliers, sd of good points is 1.5
set.seed(123)
r2 <- c(rnorm(135, sd=1.5), rnorm(15, mean=-10, sd=.5))
locScaleM(r2)
```

`logregBY` *Bianco and Yohai estimator for logistic regression*

Description

This function computes the M-estimator proposed by Bianco and Yohai for logistic regression. By default, an intercept term is included and p parameters are estimated. Modified by Yohai (2018) to take as initial estimator a weighted ML estimator with weights derived from the MCD estimator. For more details we refer to Croux, C., and Haesbroeck, G. (2002), "Implementing the Bianco and Yohai estimator for Logistic Regression"

Usage

```
logregBY(x0, y, intercept = 1, const = 0.5, kmax = 1000,
  maxhalf = 10)
```

Arguments

`x0`	matrix of explanatory variables;
`y`	vector of binomial responses (0 or 1);
`intercept`	1 or 0 indicating if an intercept is included or or not
`const`	tuning constant used in the computation of the estimator (default=0.5);
`kmax`	maximum number of iterations before convergence (default=1000);
`maxhalf`	max number of step-halving (default=10).

Value

A list with the following components:

`coefficients`	estimates for the regression coefficients
`standard.deviation`	standard deviations of the coefficients
`fitted.values`	fitted values
`residual.deviances`	residual deviances
`components`	logical value indicating whether convergence was achieved
`objective`	value of the objective function at the minimum

Author(s)

Christophe Croux, Gentiane Haesbroeck, Victor Yohai

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(skin)
Xskin <- as.matrix( skin[, 1:2] )
yskin <- skin$vasoconst
skinBY <- logregBY(Xskin, yskin, intercept=1)
skinBY$coeff
skinBY$standard.deviation
```

---

logregWBY *Bianco and Yohai estimator for logistic regression*

Description

This function computes the weighted M-estimator of Bianco and Yohai in logistic regression. By default, an intercept term is included and p parameters are estimated. Modified by Yohai (2018) to take as initial estimator a weighted ML estimator computed with weights derived from the MCD estimator of the continuous explanatory variables. The same weights are used to compute the final weighted M-estimator. For more details we refer to Croux, C., and Haesbroeck, G. (2002), "Implementing the Bianco and Yohai estimator for Logistic Regression"

Usage

```
logregWBY(x0, y, intercept = 1, const = 0.5, kmax = 1000,
  maxhalf = 10)
```

Arguments

x0	matrix of explanatory variables;
y	vector of binomial responses (0 or 1);
intercept	1 or 0 indicating if an intercept is included or or not
const	tuning constant used in the computation of the estimator (default=0.5);
kmax	maximum number of iterations before convergence (default=1000);
maxhalf	max number of step-halving (default=10).

Value

A list with the following components:

`coefficients`	estimates for the regression coefficients
`standard.deviation`	standard deviations of the coefficients
`fitted.values`	fitted values
`residual.deviances`	residual deviances
`components`	logical value indicating whether convergence was achieved
`objective`	value of the objective function at the minimum

Author(s)

Christophe Croux, Gentiane Haesbroeck, Victor Yohai

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(skin)
Xskin <- as.matrix( skin[, 1:2] )
yskin <- skin$vasoconst
skinWBY <- logregWBY(Xskin, yskin, intercept=1)
skinWBY$coeff
skinWBY$standard.deviation
```

---

logregWML *Weighted likelihood estimator for the logistic model*

Description

This function computes a weighted likelihood estimator for the logistic model, where the weights penalize high leverage observations. In this version the weights are zero or one.

Usage

```
logregWML(x0, y, intercept = 1)
```

Arguments

`x0`	p x n matrix of explanatory variables, p is the number of explanatory variables, n is the number of observations
`y`	response vector
`intercept`	1 or 0 indicating if an intercept is included or or not

Value

A list with the following components:

`coefficients`	vector of regression coefficients
`standard.deviation`	standard deviations of the regression coefficient estimators
`fitted.values`	vector with the probabilities of success
`residual.deviances`	residual deviances
`cov`	covariance matrix of the regression estimates
`objective`	value of the objective function at the minimum
`xweights`	vector of zeros and ones used to compute the weighted maimum likelihood estimator

Author(s)

Victor Yohai

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(skin)
Xskin <- as.matrix( skin[, 1:2] )
yskin <- skin$vasoconst
skinWML <- logregWML(Xskin, yskin, intercept=1)
skinWML$coeff
skinWML$standard.deviation
```

---

mineral *Mineral data*

Description

Mineral data. More details here.

Usage

```
data(mineral)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(mineral)
```

MMPY *MM regression estimator using Pen~a-Yohai candidates*

Description

This function computes MM-regression estimator using Pen~a-Yohai candidates for the initial S-estimator. This function is used internally by lmrobdetMM, and not meant to be used directly.

Usage

```
MMPY(X, y, control, mf)
```

Arguments

X	design matrix
y	response vector
control	a list of control parameters as returned by lmrobdet.control
mf	model frame

Value

an lmrob object witht the M-estimator obtained starting from the S-estimator computed with the Pen~a-Yohai initial candidates. The properties of the final estimator (efficiency, etc.) are determined by the tuning constants in the argument control.

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

modopt *Tuning parameter for a rho function in the modified (asymptotic bias-) optimal family*

Description

This function computes the tuning constant that yields an MM-regression estimator with a desired asymptotic efficiency when computed with a rho function in the corresponding family. The output of this function can be passed to the functions lmrobdet.control, mscale and rho.

Usage

```
modopt(e)
```

Arguments

e	the desired efficiency of the corresponding regression estimator for Gaussian errors

Value

A vector with named elements containing the corresponding tuning parameters.

Author(s)

Kjell Konis

Examples

```
# Tuning parameters for an 85%-efficient M-estimator at a Gaussian model
modopt(.85)
```

neuralgia *Neuralgia data*

Description

Neuralgia data. More details here.

Usage

```
data(neuralgia)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(neuralgia)
```

`oats` *Oats data*

Description

Oats data. More details here.

Usage

```
data(oats)
```

Format

An object of class `"data.frame"`.

Source

Source goes here.

References

References go here.

Examples

```
data(oats)
```

optimal — *Tuning parameter for a rho function in the (asymptotic bias-) optimal family*

Description

This function computes the tuning constant that yields an MM-regression estimator with a desired asymptotic efficiency when computed with a rho function in the corresponding family. The output of this function can be passed to the functions lmrobdet.control, mscale and rho.

Usage

```
optimal(e)
```

Arguments

e	the desired efficiency of the corresponding regression estimator for Gaussian errors

Value

A vector with named elements containing the corresponding tuning parameters.

Author(s)

Kjell Konis

Examples

```
# Tuning parameters for an 85%-efficient M-estimator at a Gaussian model
optimal(.85)
```

pcaRobS — *Robust principal components*

Description

This function computes robust principal components based on the minimization of the "residual" M-scale.

Usage

```
pcaRobS(X, ncomp, desprop = 0.9, deltasca = 0.5, maxit = 100)
```

Arguments

`X`	a data matrix with observations in rows.
`ncomp`	desired (maximum) number of components
`desprop`	desired (minimum) proportion of explained variability (default = 0.9)
`deltasca`	"delta" parameter of the scale M-estimator (default=0.5)
`maxit`	maximum number of iterations (default= 100)

Value

A list with the following components:

`q`	The actual number of principal components
`propex`	The actual proportion of unexplained variability
`eigvec`	Eigenvectors, in a `p x q` matrix
`fit`	an `n x p` matrix with the rank-q approximation to `X`
`repre`	An `n x q` matrix with representation of data in R^q (scores)
`propSPC`	A vector of length `p` with the cumulative explained variance from initial SPC

Author(s)

Ricardo Maronna, <rmaronna@retina.ar>, based on original code by D. Pen~a and J. Prieto

References

http://www.wiley.com/go/maronna/robust

Examples

```
data(bus)
X0 <- as.matrix(bus)
X1 <- X0[,-9]
ss <- apply(X1, 2, mad)
mu <- apply(X1, 2, median)
X <- scale(X1, center=mu, scale=ss)
q <- 3  #compute three components
rr <- pcaRobS(X, q, 0.99)
round(rr$eigvec, 3)
```

prcompRob *Robust Principal Components Cont'd*

Description

This function uses the pcaRobS function to compute all principal components while behaving similarly to the prcomp function

Usage

```
prcompRob(x, rank. = NULL, delta.scale = 0.5, max.iter = 100L)
```

Arguments

x	data matrix with observations in rows
rank.	Maximal number of principal components to be used (optional)
delta.scale	"delta" parametor of the scale M-estimator (default = 0.5)
max.iter	maximum number of iterations (default = 100)

Value

sdev	the standard deviation of the principal components
rotation	matrix containing the factor loadings
x	matrix containing the rotated data
center	the centering used

Author(s)

Gregory Brownson, <gregory.brownson@gmail.com>

Examples

```
data(wine)

p.wine <- prcompRob(wine)
summary(p.wine)

## Choose only 5
p5.wine <- prcompRob(wine, rank. = 5)
summary(p5.wine)
```

refine.sm *IRWLS iterations for S- or M-estimators*

Description

This function performs iterative improvements for S- or M-estimators.

Usage

```
refine.sm(x, y, initial.beta, initial.scale, k = 50, conv = 1, b, cc,
  family, step = "M")
```

Arguments

x	design matrix
y	vector of responses
initial.beta	vector of initial regression estimates
initial.scale	initial residual scale estimate. If missing the (scaled) median of the absolute residuals is used.
k	maximum number of refining steps to be performed
conv	an integer indicating whether to check for convergence (1) at each step, or to force running k steps (0)
b	tuning constant for the M-scale estimator, used if iterations are for an S-estimator.
cc	tuning constant for the rho function.
family	string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt")
step	a string indicating whether the iterations are to compute an S-estiamator ('S') or an M-estimator ('M')

Details

This function performs iterative improvements for S- or M-estimators. Both iterations are formally the same, the only difference is that for M-iterations the residual scale estimate remains fixed, while for S-iterations it is updated at each step. In this case, we follow the Fast-S algorithm of Salibian-Barrera and Yohai an use one step updates for the M-scale, as opposed to a full computation. This as internal function.

Value

A list with the following components:

beta.rw	The updated vector of regression coefficients
scale.rw	The corresponding estimated residual scale
converged	A logical value indicating whether the algorithm converged

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>.

---

resex *Resex data*

Description

Resex data. More details here.

Usage

```
data(resex)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(resex)
```

---

rho *Rho functions*

Description

This function returns the value of the "rho" loss function used to compute either an M-scale estimator or a robust regression estimator. It currently can be used to compute the bisquare, optimal and modified optimal loss functions.

Usage

```
rho(u, family = " bisquare", cc, standardize = TRUE)
```

Arguments

u	point or vector at which rho is to be evaluated
family	family string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt").
cc	tuning parameters to be computed according to efficiency and / or breakdown considerations. See lmrobdet.control, bisquare, modopt and optimal.
standardize	logical value determining whether the rho function is to be standardized so that its maximum value is 1. See Mpsi.

Value

The value(s) of rho at u

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

Examples

```
# Evaluate rho tuned for 85% efficiency
rho(u=1.1, family='bisquare', cc=bisquare(.85))
# Evaluate rho tuned for 50% breakdown
rho(u=1.1, family='optimal', cc=lmrobdet.control(bb=.5, family='optimal')$tuning.chi)
```

---

rhoprime *The first derivative of the rho function*

Description

The first derivative of the rho function

Usage

```
rhoprime(u, family, cc, standardize = FALSE)
```

Arguments

u	point or vector at which rho is to be evaluated
family	family string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt").
cc	tuning parameters to be computed according to efficiency and / or breakdown considerations. See lmrobdet.control, bisquare, modopt and optimal.
standardize	logical value determining whether the rho function is to be standardized so that its maximum value is 1. See Mpsi.

Value

The value of the first derivative rho evaluated at u

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

Examples

```
# Evaluate the derivative of a rho function tuned for 85% efficiency
rhoprime(u=1.1, family='bisquare', cc=bisquare(.85))
# Evaluate the derivative of a rho function tuned for 50% breakdown
rhoprime(u=1.1, family='optimal', cc=lmrobdet.control(bb=.5, family='optimal')$tuning.chi)
```

---

rhoprime2 *The second derivative of the rho function*

Description

The second derivative of the rho function

Usage

```
rhoprime2(u, family, cc, standardize = FALSE)
```

Arguments

u	point or vector at which rho is to be evaluated
family	family string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt").
cc	tuning parameters to be computed according to efficiency and / or breakdown considerations. See lmrobdet.control, bisquare, modopt and optimal.
standardize	logical value determining whether the rho function is to be standardized so that its maximum value is 1. See Mpsi.

Value

The value of the second derivative of rho evaluated at u

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

Examples

```
# Evaluate the 2nd derivative of a rho function tuned for 85% efficiency
rhoprime2(u=1.1, family='bisquare', cc=bisquare(.85))
# Evaluate the 2nd derivative of a rho function tuned for 50% breakdown
rhoprime2(u=1.1, family='optimal', cc=lmrobdet.control(bb=.5, family='optimal')$tuning.chi)
```

`scaleM` *M-scale estimator*

Description

This function computes an M-scale, which is a robust scale (spread) estimator. M-estimators of scale are a robust alternative to the sample standard deviation. Given a vector of residuals `r`, the M-scale estimator `s` solves the non-linear equation `mean(rho(r/s, cc))=b`, where `b` and `cc` are user-chosen tuning constants. In this package the function `rho` is one of Tukey's bisquare family. The breakdown point of the estimator is `min(b, 1-b)`, so the optimal choice for `b` is 0.5. To obtain a consistent estimator the constant `cc` should be chosen such that E(rho(Z, cc)) = b, where Z is a standard normal random variable.

Usage

```
scaleM(u, delta = 0.5, tuning.chi = 1.547645, family = "bisquare",
  max.it = 100, tol = 1e-06)
```

Arguments

`u`	vector of residuals
`delta`	the right hand side of the M-scale equation
`tuning.chi`	the tuning object for the rho function as returned by `lmrobdet.control`, bisquare, modopt or optimal. It should correspond to the family of rho functions specified in the argument `family`.
`family`	string specifying the name of the family of loss function to be used (current valid options are "bisquare", "optimal" and "modopt").
`max.it`	maximum number of iterations allowed
`tol`	relative tolerance for convergence

Details

The iterative algorithm starts from the scaled median of the absolute values of the input vector, and then cycles through the equation s^2 = s^2 * mean(rho(r/s, cc)) / b.

Value

The scale estimate value at the last iteration or at convergence.

Author(s)

Matias Salibian-Barrera, <matias@stat.ubc.ca>

Examples

```
set.seed(123)
r <- rnorm(150, sd=1.5)
mscale(r)
sd(r)
# 10% of outliers, sd of good points is 1.5
set.seed(123)
r2 <- c(rnorm(135, sd=1.5), rnorm(15, mean=-5, sd=.5))
mscale(r2)
sd(r2)
```

ShinyUI *Open the Shiny interface for the package*

Description

This function opens the Shiny interface for the package.

Usage

```
ShinyUI()
```

Author(s)

Gregory Brownson <gsb25@uw.edu>

References

http://www.wiley.com/go/maronna/robust

---

shock *Shock data*

Description

Shock data. More details here.

Usage

```
data(shock)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(shock)
```

`skin` *Skin data*

Description

Skin data. More details here.

Usage

```
data(skin)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(skin)
```

SMPY *SM regression estimator using Pen~a-Yohai candidates*

Description

This function computes a robust regression estimator when there are categorical / dummy explanatory variables. It uses Pen~a-Yohai candidates for the S-estimator. This function is used internally by lmrobdetMM, and not meant to be used directly.

Usage

```
SMPY(mf, y, control, split)
```

Arguments

mf	model frame
y	response vector
control	a list of control parameters as returned by lmrobdet.control
split	a list as returned by splitFrame containing the continuous and dummy components of the design matrix

Value

an lmrob object witht the M-estimator obtained starting from the MS-estimator computed with the Pen~a-Yohai initial candidates. The properties of the final estimator (efficiency, etc.) are determined by the tuning constants in the argument control.

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

`step.lmrobdetMM` *Robust stepwise using RFPE*

Description

This function performs stepwise model selection on a robustly fitted linear model using the RFPE criterion and the robust regression estimators computed with `lmrobdetMM`. Only backwards stepwise is currently implemented.

Usage

```
step.lmrobdetMM(object, scope, direction = c("both", "backward",
  "forward"), trace = TRUE, keep = NULL, steps = 1000,
  whole.path = FALSE)
```

Arguments

`object`	a robust fit as returned by `lmrobdetMM`
`scope`	either a formula or a list with elements `lower` and `upper` each of which is a formula. The terms in the right-hand-side of `lower` are always included in the model and the additional terms in the right-hand-side of `upper` are the candidates for inclusion/exclusion from the model. If a single formula is given, it is taken to be `upper`, and `lower` is set to the empty model. The `.` operator is interpreted in the context of the formula in `object`.
`direction`	the direction of stepwise search. Currenly only `backward` stepwise searches are implemented.
`trace`	logical. If `TRUE` information about each step is printed on the screen.
`keep`	a filter function whose input is a fitted model object and the associated AIC statistic, and whose output is arbitrary. Typically keep will select a subset of the components of the object and return them. The default is not to keep anything.
`steps`	maximum number of steps to be performed. Defaults to 1000, which should mean as many as needed.
`whole.path`	if `FALSE` (default) variables are dropped until the RFPE fails to improve. If `TRUE` the best variable to be dropped is removed, even if this does not improve the RFPE.

Details

Presently only backward stepwise selection is supported. During each step the Robust Final Prediction Error (as computed by the function `lmrobdetMM.RFPE`) is calculated for the current model and for each sub-model achievable by deleting a single term. If the argument `whole.path` is `FALSE`, the function steps to the sub-model with the lowest Robust Final Prediction Error or, if the current model has the lowest Robust Final Prediction Error, terminates. If the argument `whole.path` is `TRUE`, the function steps through all smaller submodels removing, at each step, the variable that most reduces the Robust Final Prediction Error. The scale estimate from `object` is used to compute the Robust Final Prediction Error throughout the procedure.

Value

If `whole.path == FALSE` the function returns the robust fit as obtained by `lmrobdetMM` using the final model. If `whole.path == TRUE` a list is returned containing the RFPE of each model on the sequence of submodels. The names of the components of this list are the formulas that correspods to each model.

Author(s)

Victor Yohai, <victoryohai@gmail.com>, Matias Salibian-Barrera, <matias@stat.ubc.ca>

References

http://www.wiley.com/go/maronna/robust

See Also

DCML, MMPY, SMPY

Examples

```
cont <- lmrobdet.control(bb = 0.5, efficiency = 0.85, family = "bisquare")
set.seed(300)
X <- matrix(rnorm(50*6), 50, 6)
beta <- c(1,1,1,0,0,0)
y <- as.vector(X %*% beta) + 1 + rnorm(50)
y[1:6] <- seq(30, 55, 5)
for (i in 1:6) X[i,] <- c(X[i,1:3],i/2,i/2,i/2)
Z <- cbind(y,X)
Z <- as.data.frame(Z)
obj <- lmrobdetMM(y ~ ., data=Z, control=cont)
out <- step.lmrobdetMM(obj)
```

---

vehicle *Vehicle data*

Description

Vehicle data. More details here.

Usage

```
data(vehicle)
```

Format

An object of class "data.frame".

Source

Source goes here.

References

References go here.

Examples

```
data(vehicle)
```

`wine` *Wine data*

Description

Wine data. More details here.

Usage

```
data(wine)
```

Format

An object of class `"data.frame"`.

Source

Source goes here.

References

References go here.

Examples

```
data(wine)
```

Index