

Package ‘robustbase’

May 14, 2012

Version 0.9-1

Date 2012-05-14

Title Basic Robust Statistics

Author Original code by many authors, notably Peter Rousseeuw and Christophe Croux, see file ‘Copyrights’; Valentin Todorov <valentin.todorov@chello.at>, Andreas Ruckstuhl <andreas.ruckstuhl@zhaw.ch>, Matias Salibian-Barrera <matias@stat.ubc.ca>, Tobias Verbeke <tobias.verbeke@openanalytics.eu>, Manuel Koller <koller@stat.math.ethz.ch>, Martin Maechler

Maintainer Martin Maechler <maechler@stat.math.ethz.ch>

URL <http://robustbase.r-forge.r-project.org/>

Description “Essential” Robust Statistics. The goal is to provide tools allowing to analyze data with robust methods. This includes regression methodology including model selections and multivariate statistics where we strive to cover the book “Robust Statistics, Theory and Methods” by Maronna, Martin and Yohai; Wiley 2006.

Depends R (>= 2.13.0), stats, graphics, methods

Suggests grid, MASS, lattice, boot, cluster, Matrix, MPV, xtable, ggplot2, RColorBrewer, reshape2

LazyData yes

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-05-14 15:13:27

R topics documented:

adjbox	4
adjboxStats	7
adjOutlyingness	8
aircraft	10
airmay	11
alcohol	12
ambientNOxCH	13
Animals2	16
anova.glmrob	17
anova.lmrob	19
bushfire	21
carrots	22
chgDefaults-methods	23
cloud	23
coleman	24
condroz	25
covMcd	26
covOGK	28
CrohnD	31
cushny	32
delivery	33
education	34
epilepsy	35
exAM	36
functionX-class	37
functionXal-class	38
glmrob	38
glmrobMqle.control	43
h.alpha.n	44
hbk	45
heart	46
huberM	47
kootenay	48
lactic	49
lmrob	50
lmrob.D.fit	53
lmrob.M.fit	55
lmrob.control	57
lmrob.fit	60
lmrob.lar	61
lmrob.M.S	62
lmrob.S	64
los	66
ltsReg	67
mc	70
milk	72

nlrob	73
NOxEmissions	76
pension	78
phosphor	79
pilot	79
plot.lmrob	80
plot.lts	81
plot.mcd	83
possumDiv	85
predict.glmrob	87
predict.lmrob	89
print.lmrob	90
psiFunc	91
psi_func-class	92
pulpfiber	93
Qn	94
radarImage	95
residuals.glmrob	96
rrcov.control	97
salinity	99
scaleTau2	100
SiegelsEx	101
Sn	102
splitFrame	103
starsCYG	105
summarizeRobWeights	106
summary.glmrob	107
summary.lmrob	108
summary.lts	109
summary.mcd	111
summary.nlrob	112
telef	113
tolEllipsePlot	114
toxicity	115
tukeyChi	116
tukeyPsi1	117
vaso	118
wagnerGrowth	119
wgt.himedian	121
wood	121

adjbox

*Plot an Adjusted Boxplot for Skew Distributions***Description**

Produces boxplots adjusted for skewed distributions as proposed in Hubert and Vandervieren (2004).

Usage

```
adjbox(x, ...)

## S3 method for class 'formula'
adjbox(formula, data = NULL, ..., subset, na.action = NULL)

## Default S3 method:
adjbox(x, ..., range = 1.5, doReflect = FALSE,
       width = NULL, varwidth = FALSE,
       notch = FALSE, outline = TRUE, names, plot = TRUE,
       border = par("fg"), col = NULL, log = "",
       pars = list(boxwex = 0.8, staplewex = 0.5, outwex = 0.5),
       horizontal = FALSE, add = FALSE, at = NULL)
```

Arguments

formula	a formula, such as $y \sim \text{grp}$, where y is a numeric vector of data values to be split into groups according to the grouping variable grp (usually a factor).
data	a data.frame (or list) from which the variables in formula should be taken.
subset	an optional vector specifying a subset of observations to be used for plotting.
na.action	a function which indicates what should happen when the data contain NAs. The default is to ignore missing values in either the response or the group.
x	for specifying data from which the boxplots are to be produced. Either a numeric vector, or a single list containing such vectors. Additional unnamed arguments specify further data as separate vectors (each corresponding to a component boxplot). NAs are allowed in the data.
...	For the formula method, named arguments to be passed to the default method. For the default method, unnamed arguments are additional data vectors (unless x is a list when they are ignored), and named arguments are arguments and graphical parameters to be passed to <code>bxp</code> in addition to the ones given by argument <code>pars</code> (and override those in <code>pars</code>).
range	this determines how far the plot whiskers extend out from the box, and is simply passed as argument <code>coef</code> to <code>adjboxStats()</code> . If <code>range</code> is positive, the whiskers extend to the most extreme data point which is no more than <code>range</code> times the interquartile range from the box. A value of zero causes the whiskers to extend to the data extremes.

doReflect	logical indicating if the MC should also be computed on the <i>reflected</i> sample $-x$, and be averaged, see <code>mc</code> .
width	a vector giving the relative widths of the boxes making up the plot.
varwidth	if varwidth is TRUE, the boxes are drawn with widths proportional to the square-roots of the number of observations in the groups.
notch	if notch is TRUE, a notch is drawn in each side of the boxes. If the notches of two plots do not overlap this is ‘strong evidence’ that the two medians differ (Chambers <i>et al.</i> , 1983, p. 62). See <code>boxplot.stats</code> for the calculations used.
outline	if outline is not true, the outliers are not drawn (as points whereas S+ uses lines).
names	group labels which will be printed under each boxplot.
boxwex	a scale factor to be applied to all boxes. When there are only a few groups, the appearance of the plot can be improved by making the boxes narrower.
staplewex	staple line width expansion, proportional to box width.
outwex	outlier line width expansion, proportional to box width.
plot	if TRUE (the default) then a boxplot is produced. If not, the summaries which the boxplots are based on are returned.
border	an optional vector of colors for the outlines of the boxplots. The values in border are recycled if the length of border is less than the number of plots.
col	if col is non-null it is assumed to contain colors to be used to colour the bodies of the box plots. By default they are in the background colour.
log	character indicating if x or y or both coordinates should be plotted in log scale.
pars	a list of (potentially many) more graphical parameters, e.g., boxwex or outpch; these are passed to <code>bxp</code> (if plot is true); for details, see there.
horizontal	logical indicating if the boxplots should be horizontal; default FALSE means vertical boxes.
add	logical, if true <i>add</i> boxplot to current plot.
at	numeric vector giving the locations where the boxplots should be drawn, particularly when add = TRUE; defaults to 1:n where n is the number of boxes.

Details

The generic function `adjbox` currently has a default method (`adjbox.default`) and a formula interface (`adjbox.formula`).

If multiple groups are supplied either as multiple arguments or via a formula, parallel boxplots will be plotted, in the order of the arguments or the order of the levels of the factor (see `factor`).

Missing values are ignored when forming boxplots.

Extremes of the upper and whiskers of the adjusted boxplots are computed using the `medcouple` (`mc()`), a robust measure of skewness. For details, cf. `TODO`

Value

A `list` with the following components:

<code>stats</code>	a matrix, each column contains the extreme of the lower whisker, the lower hinge, the median, the upper hinge and the extreme of the upper whisker for one group/plot. If all the inputs have the same class attribute, so will this component.
<code>n</code>	a vector with the number of observations in each group.
<code>coef</code>	a matrix where each column contains the lower and upper extremes of the notch.
<code>out</code>	the values of any data points which lie beyond the extremes of the whiskers.
<code>group</code>	a vector of the same length as <code>out</code> whose elements indicate to which group the outlier belongs.
<code>names</code>	a vector of names for the groups.

Note

The code and documentation only slightly modifies the code of `boxplot.default`, `boxplot.formula` and `boxplot.stats`

Author(s)

R Core Development Team, slightly adapted by Tobias Verbeke

References

Vandervieren, E., Hubert, M. (2004) An adjusted boxplot for skewed distributions, in: Antoch, J., ed. (2004). *Proceedings in Computational Statistics 2004*; Heidelberg: Springer-Verlag, 1933–1940.

Hubert, M. and Vandervieren, E. (2006) *An Adjusted Boxplot for Skewed Distributions*, Technical Report TR-06-11, KU Leuven, Section of Statistics, Leuven.

<http://wis.kuleuven.be/stat/robust/Papers/TR0611.pdf>

See Also

The `medcouple`, `mc`; `boxplot`.

Examples

```
if(getRversion() >= "2.9.0" && require("boot")) {
  ### Hubert and Vandervieren (2006), p. 10, Fig. 4.
  data(coal, package = "boot")
  coaldiff <- diff(coal$date)
  op <- par(mfrow = c(1,2))
  boxplot(coaldiff, main = "Original Boxplot")
  adjbox(coaldiff, main = "Adjusted Boxplot")
  par(op)
}

### Hubert and Vandervieren (2006), p. 11, Fig. 6. -- enhanced
```

```

op <- par(mfrow = c(2,2), mar = c(1,3,3,1), oma = c(0,0,3,0))
with(condroz, {
  boxplot(Ca, main = "Original Boxplot")
  adjbox (Ca, main = "Adjusted Boxplot")
  boxplot(Ca, main = "Original Boxplot [log]", log = "y")
  adjbox (Ca, main = "Adjusted Boxplot [log]", log = "y")
})
mtext("'Ca' from data(condroz)",
      outer=TRUE, font = par("font.main"), cex = 2)
par(op)

```

adjboxStats

Statistics for Skewness-adjusted Boxplots

Description

Computes the “statistics” for producing boxplots adjusted for skewed distributions as proposed in Hubert and Vandervieren (2004), see [adjbox](#).

Usage

```
adjboxStats(x, coef = 1.5, a = -4, b = 3, do.conf = TRUE, do.out = TRUE,
           ...)
```

Arguments

x a numeric vector for which adjusted boxplot statistics are computed.

coef number determining how far ‘whiskers’ extend out from the box, see [boxplot.stats](#).

a, b scaling factors multiplied by the medcouple [mc\(\)](#) to determine outlier boundaries; see the references.

do.conf, do.out logicals; if FALSE, the conf or out component respectively will be empty in the result.

... further optional arguments to be passed to [mc\(\)](#), such as `doReflect`.

Details

Given the quartiles Q_1, Q_3 , the interquartile range $\Delta Q := Q_3 - Q_1$, and the medcouple $M := mc(x)$, $c = coef$, the “fence” is defined, for $M \geq 0$ as

$$[Q_1 - ce^{a \cdot M} \Delta Q, Q_3 + ce^{b \cdot M} \Delta Q],$$

and for $M < 0$ as

$$[Q_1 - ce^{-b \cdot M} \Delta Q, Q_3 + ce^{-a \cdot M} \Delta Q],$$

and all observations x outside the fence, the “potential outliers”, are returned in `out`.

Note that a typo in robustbase version up to 0.7-8, for the (rare left-skewed) case where $mc(x) < 0$, lead to a “fence” not wide enough in the upper part, and hence *less* outliers there.

Value

A [list](#) with the components

stats	a vector of length 5, containing the extreme of the lower whisker, the lower hinge, the median, the upper hinge and the extreme of the upper whisker.
n	the number of observations
conf	the lower and upper extremes of the ‘notch’ (if(do.conf)). See boxplot.stats .
fence	length 2 vector of interval boundaries which define the non-outliers, and hence the whiskers of the plot.
out	the values of any data points which lie beyond the fence, and hence beyond the extremes of the whiskers.

Note

The code only slightly modifies the code of R’s [boxplot.stats](#).

Author(s)

R Core Development Team ([boxplot.stats](#)); adapted by Tobias Verbeke and Martin Maechler.

See Also

[adjbox\(\)](#), also for references, the function which mainly uses this one; further [boxplot.stats](#).

Examples

```
data(condroz)
adjboxStats(ccA <- condroz[, "Ca"])
adjboxStats(ccA, doReflect = TRUE)# small difference in fence

## Test reflection invariance [was not ok, up to and including robustbase_0.7-8]
a1 <- adjboxStats( ccA, doReflect = TRUE)
a2 <- adjboxStats(-ccA, doReflect = TRUE)

nm1 <- c("stats", "conf", "fence")
stopifnot(all.equal(      a1[nm1],
                      lapply(a2[nm1], function(u) rev(-u))),
          all.equal(a1[["out"]], -a2[["out"]]))
```

adjOutlyingness

Compute Skewness-adjusted Multivariate Outlyingness

Description

For an $n \times p$ data matrix (or data frame) x , compute the “outlyingness” of all n observations. Outlyingness here is a generalization of the Donoho-Stahel outlyingness measure, where skewness is taken into account via the medcouple, [mc\(\)](#).

Usage

```
adjOutlyingness(x, ndir = 250, clower = 3, cupper = 4,
               alpha.cutoff = 0.75, coef = 1.5, qr.tol = 1e-12)
```

Arguments

x	a numeric <code>matrix</code> or <code>data.frame</code> .
ndir	positive integer specifying the number of directions that should be searched.
clower, cupper	the constant to be used for the lower and upper tails, in order to transform the data towards symmetry.
alpha.cutoff	number in (0,1) specifying the quantiles $(\alpha, 1-\alpha)$ which determine the “outlier” cutoff.
coef	positive number specifying the factor with which the interquartile range (IQR) is multiplied to determine ‘boxplot hinges’-like upper and lower bounds.
qr.tol	positive tolerance to be used for <code>qr</code> and <code>solve.qr</code> for determining the <code>ndir</code> directions, each determined by a random sample of p (out of n) observations.

Details

FIXME: Details in the comment of the Matlab code; also in the reference(s).

The method as described can be useful as preprocessing in FASTICA (<http://www.cis.hut.fi/projects/ica/fastica/>); see also the R package **fastICA**.

Value

a list with components	
adjout	numeric of length(n) giving the adjusted outlyingness of each observation.
cutoff	cutoff for “outlier” with respect to the adjusted outlyingnesses, and depending on <code>alpha.cutoff</code> .
nonOut	logical of length(n), TRUE when the corresponding observation is non -outlying with respect to the cutoff and the adjusted outlyingnesses.

Note

The result is *random* as it depends on the sample of `ndir` directions chosen.

Author(s)

Guy Brys; help page and improvements by Martin Maechler

References

Brys, G., Hubert, M., and Rousseeuw, P.J. (2005) A Robustification of Independent Component Analysis; *Journal of Chemometrics*, **19**, 1–12.

For the up-to-date reference, please consult <http://wis.kuleuven.be/stat/robust.html>

See Also

the adjusted boxplot, [adjbox](#) and the [medcouple](#), [mc](#).

Examples

```
## An Example with bad condition number and "border case" outliers

if(FALSE) {## Not yet ok, because of bug in adjOutl
  dim(longley)
  set.seed(1) ## result is random %% and there's a bug - FIXME! -- try set.seed(3)
  ao1 <- adjOutlyingness(longley)
  ## which are not outlying ?
  table(ao1$nonOut) ## all of them
  stopifnot(all(ao1$nonOut))
}

## An Example with outliers :

dim(hbk)
set.seed(1)
ao.hbk <- adjOutlyingness(hbk)
str(ao.hbk)
hist(ao.hbk $adjout)## really two groups
table(ao.hbk$nonOut)## 14 outliers, 61 non-outliers:
## outliers are :
which(! ao.hbk$nonOut) # 1 .. 14 --- but not for all random seeds!

## here, they are the same as found by (much faster) MCD:
cc <- covMcd(hbk)
stopifnot(all(cc$mcd.wt == ao.hbk$nonOut))

## This is revealing (about 1--2 cases, where outliers are *not* == 1:14
## but needs almost 1 [sec] per call:
if(interactive()) {
  for(i in 1:30) {
    print(system.time(ao.hbk <- adjOutlyingness(hbk)))
    if(!identical(iout <- which(!ao.hbk$nonOut), 1:14)) {
      cat("Outliers:\n"); print(iout)
    }
  }
}
```

aircraft

Aircraft Data

Description

Aircraft Data, deals with 23 single-engine aircraft built over the years 1947-1979, from Office of Naval Research. The dependent variable is cost (in units of \$100,000) and the explanatory variables are aspect ratio, lift-to-drag ratio, weight of plane (in pounds) and maximal thrust.

Usage

```
data(aircraft)
```

Format

A data frame with 23 observations on the following 5 variables.

X1 Aspect Ratio

X2 Lift-to-Drag Ratio

X3 Weight

X4 Thrust

Y Cost

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, page 154, table 22.

Examples

```
data(aircraft)
summary( lm.airc <- lm(Y ~ ., data = aircraft))
summary(rlm.airc <- MASS::rlm(Y ~ ., data = aircraft))

aircraft.x <- data.matrix(aircraft[,1:4])
c_air <- covMcd(aircraft.x)
c_air
```

airmay

Air Quality Data

Description

Air Quality Data Set for May 1973, from Chambers et al. (1983). The whole data set consists of daily readings of air quality values from May 1, 1973 to September 30, 1973, but here are included only the values for May. This data set is an example of the special treatment of the missing values.

Usage

```
data(airmay)
```

Format

A data frame with 31 observations on the following 4 variables.

- X1 Solar Radiation in Longleys in the frequency band 4000-7700 from 0800 to 1200 hours at Central Park
- X2 Average windspeed (in miles per hour) between 7000 and 1000 hours at La Guardia Airport
- X3 Maximum daily temperature (in degrees Fahrenheit) at La Guardia Airport
- Y Mean ozone concentration (in parts per billion) from 1300 to 1500 hours at Roosevelt Island

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.86, table 6.

Examples

```
data(airmay)
summary(lm.airmay <- lm(Y ~ ., data=airmay))

airmay.x <- data.matrix(airmay[,1:3])
```

alcohol

Alcohol Solubility in Water Data

Description

The solubility of alcohols in water is important in understanding alcohol transport in living organisms. This dataset from (Romanelli et al., 2001) contains physicochemical characteristics of 44 aliphatic alcohols. The aim of the experiment was the prediction of the solubility on the basis of molecular descriptors.

Usage

```
data(alcohol)
```

Format

A data frame with 44 observations on the following 7 numeric variables.

- SAG solvent accessible surface-bounded molecular volume.
- V volume
- logPC Log(PC); PC = octanol-water partitions coefficient
- P polarizability
- RM molar refractivity
- Mass the mass
- logSolubility ln(Solubility), the response.

Source

The website accompanying the MMY-book: http://www.wiley.com/legacy/wileychi/robust_statistics

References

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

Examples

```
data(alcohol)
## version of data set with trivial names, as
s.alcohol <- alcohol
names(s.alcohol) <- paste("Col", 1:7, sep='')
```

ambientNOxCH

Daily Means of NOx (mono-nitrogen oxides) in air

Description

This dataset contains daily means (from midnight to midnight) of NOx, i.e., mono-nitrogen oxides, in [ppb] at 13 sites in central Switzerland and Aarau for the year 2004.

Usage

```
data(ambientNOxCH)
```

Format

A data frame with 366 observations on the following 14 variables.

date date of day, of class "Date".

ad Site is located north of Altdorf 100 meters east of motorway A2, on an open field at the beginning of a more than 2000m deep valley (690.175, 193.55; 438; inLuft)

ba Site is located in the centre of the little town of Baden in a residential area. Baden has 34'000 inhabitants and is situated on the swiss plateau (666.075, 257.972; 377; inLuft).

ef Site is located 6 km south of altdorf and 800 m north of the village of Erstfeld. The motorway A2 passes 5 m west of the measuring site. Over 8 million vehicles have passed Erstfeld in 2004 where 13% of the counts were attributed to trucks (691.43, 187.69; 457; MFM-U).

1a Site is located on a wooded hill in a rural area called Laegern, about 190 m above Baden, which is about 5 km away (669.8, 259; 690; NABEL).

1u Site is located in the center of town of Lucerne, which has 57'000 inhabitants (666.19, 211.975; 460; inLuft).

re Site is located 1 km west of Reiden on the Swiss plateau. The motorway A2 passes 5 m west of the measuring site (639.56, 232.11; 462; MFM-U).

- ri Site is located at Rigi Seebodenalp, 649 m above the lake of Lucerne on an alp with half a dozen small houses (677.9, 213.5; 1030; NABEL).
- se Site is located in Sedel next to town of Lucerne 35m above and 250m south of motorway A14 from Zug to Lucerne on a low hill with free 360° panorama (665.5, 213.41; 484; inLuft).
- si Site is located at the border of a small industrial area in Sisseln, 300 m east of a main road (640.725, 266.25; 305; inLuft).
- st Site is located at the south east border of Stans with 7'000 inhabitants (670.85, 201.025; 438; inLuft).
- su Site is located in the center of Suhr (8700 inhabitants), 10 m from the main road (648.49, 246.985; 403; inLuft).
- sz Site is located in Schwyz (14'200 inhabitants) near a shopping center (691.92, 208.03; 470; inLuft).
- zg Site is located in the centre of Zug with 22'000 inhabitants, 24 m from the main road (681.625, 224.625; 420; inLuft).

Details

The 13 sites are part of one of the three air quality monitoring networks: inLuft (regional authorities of central Switzerland and canton Aargau)

NABEL (Swiss federal network)

MFU-U (Monitoring flankierende Massnahmen Umwelt), special Swiss federal network along transit motorways A2 and A13 from Germany to Italy through Switzerland

The information within the brackets means: Swiss coordinates km east, km north; m above sea level; network

When the measuring sites are exposed to the same atmospheric condition and when there is no singular emission event at any site, $\log(\text{mean}(\text{NO}_x))$ of a specific day at each site) is a linear function of $\log(\text{yearly.mean}(\text{NO}_x))$ at the corresponding site). The offset and the slope of the straight line reflects the atmospheric conditions at this specific day. During winter time, often an inversion prevents the emissions from being diluted vertically, so that there evolve two separate atmospheric compartments: One below the inversion boundary with polluted air and one above with relatively clean air. In our example below, Rigi Seebodenalp is above the inversion boundary between December 10th and 12th.

Source

<http://www.in-luft.ch/>

http://www.empa.ch/plugin/template/empa/*:6794

<http://www.bafu.admin.ch/umweltbeobachtung/02272/02280>

See Also

another NOx dataset, [NOxEmissions](#).

Examples

```
data(ambientNOxCH)
```

```
str (ambientNOxCH)
```

```

yearly <- log(colMeans(ambientNOxCH[, -1], na.rm=TRUE))
xlim <- range(yearly)
lNOx <- log(ambientNOxCH[, -1])
days <- ambientNOxCH[, "date"]

## Subset of 9 days starting at April 4:
idays <- seq(which(ambientNOxCH$date=="2004-12-04"), length=9)
ylim <- range(lNOx[idays,], na.rm=TRUE)
op <- par(mfrow=c(3,3), mar=rep(1,4), oma = c(0,0,2,0))

for (id in idays) {
  daily <- unlist(lNOx[id,])
  plot(NA, xlim=xlim, ylim=ylim, ann=FALSE, type = "n")
  abline(0:1, col="light gray")
  abline(lmrob(daily~yearly, na.action=na.exclude),
         col="red", lwd=2)
  text(yearly, daily, names(yearly), col="blue")
  mtext(days[id], side=1, line=-1.2, cex=.75, adj=.98)
}
mtext("Daily ~ Yearly log( NOx mean values ) at 13 Swiss locations",
      outer=TRUE)
par(op)

## do all 366 regressions: Least Squares and Robust:
LS <- lapply(1:nrow(ambientNOxCH), function(id)
            lm(unlist(lNOx[id,]) ~ yearly,
              na.action = na.exclude))
R <- lapply(1:nrow(ambientNOxCH),
           function(id) lmrob(unlist(lNOx[id,]) ~ yearly,
                             na.action = na.exclude))

## currently 4 warnings about non-convergence;
## which ones?
days[notOk <- ! sapply(R, "[[", "converged") ]
## "2004-01-10" "2004-05-12" "2004-05-16" "2004-11-16"

## first problematic case:
daily <- unlist(lNOx[which(notOk)[1],])
plot(daily ~ yearly,
     main = paste("lmrob() non-convergent:", days[notOk[1]]))
rr <- lmrob(daily ~ yearly, na.action = na.exclude,
           control = lmrob.control(trace=3, max.it = 100))
##-> 53 iter.

## Look at all coefficients:
R.cf <- t(sapply(R, coef))
C.cf <- t(sapply(LS, coef))
plot(C.cf, xlim=range(C.cf[,1], R.cf[,1]),
     ylim=range(C.cf[,2], R.cf[,2]))
mD1 <- rowMeans(abs(C.cf - R.cf))
lrg <- mD1 > quantile(mD1, 0.80)
arrows(C.cf[lrg,1], C.cf[lrg,2],
       R.cf[lrg,1], R.cf[lrg,2], length=.1, col="light gray")

```

```

points(R.cf, col=2)

## All robustness weights
aW <- t(sapply(R, weights))
colnames(aW) <- names(yearly)
summary(aW)
sort(colSums(aW < 0.05, na.rm = TRUE)) # how often "clear outlier":
# lu st zg ba se sz su si re la ef ad ri
# 0 0 0 1 1 1 2 3 4 10 14 17 48

lattice::levelplot(aW, asp=1/2, main="Robustness weights",
                   xlab= "day", ylab= "site")

```

Animals2

Brain and Body Weights for 65 Species of Land Animals

Description

A data frame with average brain and body weights for 62 species of land mammals and three others.

Note that this is simply the union of [Animals](#) and [mammals](#).

Usage

```
Animals2
```

Format

body body weight in kg

brain brain weight in g

Note

After loading the **MASS** package, the data set is simply constructed by `Animals2 <- local({D <- rbind(Animals, mammal); unique(D[order(D$body, D$brain),])})`.

Rousseeuw and Leroy (1987)'s 'brain' data is the same as **MASS**'s `Animals` (with `Rat` and `Brachiosaurus` interchanged, see the example below).

Source

Weisberg, S. (1985) *Applied Linear Regression*. 2nd edition. Wiley, pp. 144–5.

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*. Wiley, p. 57.

References

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S*. Forth Edition. Springer.

Examples

```

data(Animals2)
## Sensible Plot needs doubly logarithmic scale
plot(Animals2, log = "xy")

## Regression example plot:
plotbb <- function(bbdatt) {
  d.name <- deparse(substitute(bbdatt))
  plot(log(brain) ~ log(body), data = bbdatt, main = d.name)
  abline(lm(log(brain) ~ log(body), data = bbdatt))
  abline(MASS::rlm(log(brain) ~ log(body), data = bbdatt), col = 2)
  legend("bottomright", leg = c("lm", "rlm"), col=1:2, lwd=1, inset = 1/20)
}
plotbb(bbdatt = Animals2)

## The 'same' plot for Rousseeuw's subset:
data(Animals, package = "MASS")
brain <- Animals[c(1:24, 26:25, 27:28),]
plotbb(bbdatt = brain)

lbrain <- log(brain)
plot(mahalanobis(lbrain, colMeans(lbrain), var(lbrain)),
     main = "Classical Mahalanobis Distances")
mcd <- covMcd(lbrain)
plot(mahalanobis(lbrain, mcd$center, mcd$cov),
     main = "Robust (MCD) Mahalanobis Distances")

```

 anova.glmrob

Analysis of Robust Quasi-Deviance for "glmrob" Objects

Description

Compute an analysis of robust quasi-deviance table for one or more generalized linear models fitted by [glmrob](#).

Usage

```

## S3 method for class 'glmrob'
anova(object, ..., test = c("Wald", "QD", "QDapprox"))

```

Arguments

`object, ...` objects of class `glmrob`, typically the result of a call to [glmrob](#).

`test` a character string specifying the test statistic to be used. (Partially) matching one of "Wald", "QD" or "QDapprox". See Details.

Details

Specifying a single object gives a sequential analysis of robust quasi-deviance table for that fit. That is, the reductions in the robust residual quasi-deviance as each term of the formula is added in turn are given in as the rows of a table. (*Currently not yet implemented.*)

If more than one object is specified, the table has a row for the residual quasi-degrees of freedom (However, this information is never used in the asymptotic tests). For all but the first model, the change in degrees of freedom and robust quasi-deviance is also given. (This only makes statistical sense if the models are nested.) It is conventional to list the models from smallest to largest, but this is up to the user.

In addition, the table will contain test statistics and P values comparing the reduction in robust quasi-deviance for the model on the row to that on top of it. For all robust fitting methods, the “Wald”-type test between two models can be applied (`test = "Wald"`).

When using Mallows or Huber type robust estimators (`method="Mqle"` in `glmrob`), then there are additional test methods. One is the robust quasi-deviance test (`test = "QD"`), as described by Cantoni and Ronchetti (2001). The asymptotic distribution is approximated by a chi-square distribution. Another test (`test = "QDapprox"`) is based on a quadratic approximation of the robust quasi-deviance test statistic. Its asymptotic distribution is chi-square (see the reference).

The comparison between two or more models by `anova.glmrob` will only be valid if they are fitted to the same dataset and by the same robust fitting method using the same tuning constant c (`tcc` in `glmrob`).

Value

Basically, an object of class `anova` inheriting from class `data.frame`.

Author(s)

Andreas Ruckstuhl

References

E. Cantoni and E. Ronchetti (2001) Robust Inference for Generalized Linear Models. *JASA* **96** (455), 1022–1030.

E.Cantoni (2004) Analysis of Robust Quasi-deviances for Generalized Linear Models. *Journal of Statistical Software* **10**, <http://www.jstatsoft.org/v10/i04>

See Also

`glmrob`, `anova`.

Examples

```
## Binomial response -----
data(carrots)
Cfit2 <- glmrob(cbind(success, total-success) ~ logdose + block,
               family=binomial, data=carrots, method="Mqle",
               control=glmrobMqle.control(tcc=1.2))
summary(Cfit2)
```

```

Cfit4 <- glmrob(cbind(success, total-success) ~ logdose * block,
               family=binomial, data=carrots, method="Mqle",
               control=glmrobMqle.control(tcc=1.2))

anova(Cfit2, Cfit4, test="Wald")

anova(Cfit2, Cfit4, test="QD")

anova(Cfit2, Cfit4, test="QDapprox")

## Poisson response -----
data(epilepsy)

Efit2 <- glmrob(Ysum ~ Age10 + Base4*Trt, family=poisson, data=epilepsy,
               method="Mqle", control=glmrobMqle.control(tcc=1.2,maxit=100))
summary(Efit2)

Efit3 <- glmrob(Ysum ~ Age10 + Base4 + Trt, family=poisson, data=epilepsy,
               method="Mqle", control=glmrobMqle.control(tcc=1.2,maxit=100))

anova(Efit3, Efit2, test = "Wald")

anova(Efit3, Efit2, test = "QD")

anova(Efit3, Efit2, test = "QDapprox")

```

anova.lmrob

Analysis of Robust Deviances ('anova') for 'lmrob' Objects

Description

Compute an analysis of robust Wald-type or deviance-type test tables for one or more linear regression models fitted by [lmrob](#).

Usage

```

## S3 method for class 'lmrob'
anova(object, ..., test = c("Wald", "Deviance"))

```

Arguments

object, ...	objects of class "lmrob", typically the result of a call to lmrob arguments may also be symbolic descriptions of the reduced models (cf. argument formula in lm).
test	a character string specifying the test statistic to be used. Can be one of "Wald" or "Deviance", with partial matching allowed, for specifying a "Wald"-type test or "Deviance"-type test.

Details

Specifying a single object gives a sequential analysis of a robust quasi-deviance table for that fit. That is, the reductions in the robust residual deviance as each term of the formula is added in turn are given in as the rows of a table. (Currently not yet implemented.)

If more than one object is specified, the table has a row for the residual quasi-degrees of freedom (however, this information is never used in the asymptotic tests). For all but the first model, the change in degrees of freedom and robust deviance is also given. (This only makes statistical sense if the models are nested.) As opposed to the convention, the models are forced to be listed from largest to smallest due to computational reasons.

In addition, the table will contain test statistics and P values comparing the reduction in robust deviances for the model on the row to that on top of it. There are two different robust tests available: The "Wald"-type test (`test = "Wald"`) and the Deviance-type test (`test = "Deviance"`). When using formula description of the nested models in the dot arguments and `test = "Deviance"`, you may be urged to supply a `lmrob` fit for these models by an error message. This happens when the coefficients of the largest model reduced to the nested models result in invalid initial estimates for the nested models (indicated by robustness weights which are all 0).

The comparison between two or more models by `anova.lmrob` will only be valid if they are fitted to the same dataset.

Value

Basically, an object of class `anova` inheriting from class `data.frame`.

Author(s)

Andreas Ruckstuhl

See Also

`lmrob`, `anova`.

Examples

```
data(salinity)
summary(m0.sali <- lmrob(Y ~ . , data = salinity))
anova(m0.sali, Y ~ X1 + X3)
## -> X2 is not needed
(m1.sali <- lmrob(Y ~ X1 + X3, data = salinity))
anova(m0.sali, m1.sali) # the same as before
anova(m0.sali, m1.sali, test = "Deviance")
## whereas 'X3' is highly significant:
m2 <- update(m0.sali, ~ . -X3)
anova(m0.sali, m2)
anova(m0.sali, m2, test = "Deviance")

if(require("MPV")) { ## Montgomery, Peck & Vining datasets
  Jet <- table.b13
  Jet.rflm1 <- lmrob(y ~ ., data=Jet,
                    control = lmrob.control(max.it = 500))
}
```

```
summary(Jet.rflm1)

anova(Jet.rflm1, y ~ x1 + x5 + x6, test="Wald")

try( anova(Jet.rflm1, y ~ x1 + x5 + x6, test="Deviance") )
## -> Error in anovaLm.... Please fit the nested models by lmrob

## {{ since all robustness weights become 0 in the nested model ! }}

## Ok: Do as the error message told us:
## test by comparing the two *fitted* models:

Jet.rflm2 <- lmrob(y ~ x1 + x5 + x6, data=Jet,
                  control=lmrob.control(max.it=100))
anova(Jet.rflm1, Jet.rflm2, test="Deviance")

} # end{"MPV" data}
```

bushfire

Campbell Bushfire Data

Description

This data set was used by Campbell (1984) to locate bushfire scars. The dataset contains satellite measurements on five frequency bands, corresponding to each of 38 pixels.

Usage

```
data(bushfire)
```

Format

A data frame with 38 observations on 5 variables.

Source

Maronna, R.A. and Yohai, V.J. (1995) The Behaviour of the Stahel-Donoho Robust Multivariate Estimator. *Journal of the American Statistical Association* **90**, 330–341.

Examples

```
data(bushfire)
plot(bushfire)
covMcd(bushfire)
```

 carrots

Insect Damages on Carrots

Description

The damage carrots data set from Phelps (1982) was used by McCullagh and Nelder (1989) in order to illustrate diagnostic techniques because of the presence of an outlier. In a soil experiment trial with three blocks, eight levels of insecticide were applied and the carrots were tested for insect damage.

Usage

```
data(carrots)
```

Format

A data frame with 24 observations on the following 4 variables.

success integer giving the number of carrots with insect damage.

total integer giving the total number of carrots per experimental unit.

logdose a numeric vector giving log(dose) values (eight different levels only).

block factor with levels B1 to B3

Source

Phelps, K. (1982). Use of the complementary log-log function to describe doseresponse relationships in insecticide evaluation field trials.

In R. Gilchrist (Ed.), *Lecture Notes in Statistics, No. 14. GLIM.82: Proceedings of the International Conference on Generalized Linear Models*; Springer-Verlag.

References

McCullagh P. and Nelder, J. A. (1989) *Generalized Linear Models*. London: Chapman and Hall.

Eva Cantoni and Elvezio Ronchetti (2001); JASA, and

Eva Cantoni (2004); JSS, see [glmrob](#)

Examples

```
data(carrots)
str(carrots)
plot(success/total ~ logdose, data = carrots, col = as.integer(block))
coplot(success/total ~ logdose | block, data = carrots)

## Classical glm
Cfit0 <- glm(cbind(success, total-success) ~ logdose + block,
            data=carrots, family=binomial)
summary(Cfit0)

## Robust Fit (see help(glmrob)) ....
```

chgDefaults-methods *Methods for Function chgDefaults() in Package ‘robustbase’*

Description

This is auxiliary functionality for modifying `psi_func-class` objects.

Methods

`object = "psi_func"` The method is used to change the default values for the tuning parameters, and returns a `psi_func-class` object, a copy of input object with the slot `tDefs` possibly changed;.

Examples

```
## FIXME -- add one %% --> ../experi-psi-rho-funs.R
```

cloud *Cloud point of a Liquid*

Description

This data set contains the measurements concerning the cloud point of a Liquid, from Draper and Smith (1969). The cloud point is a measure of the degree of crystallization in a stock.

Usage

```
data(cloud)
```

Format

A data frame with 19 observations on the following 2 variables.

Percentage Percentage of I-8

CloudPoint Cloud point

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.96, table 10.

Examples

```
data(cloud)
summary(lm.cloud <- lm(CloudPoint ~., data=cloud))
```

`coleman`*Coleman Data Set*

Description

Contains information on 20 Schools from the Mid-Atlantic and New England States, drawn from a population studied by Coleman et al. (1966). Mosteller and Tukey (1977) analyze this sample consisting of measurements on six different variables, one of which will be treated as a response.

Usage

```
data(coleman)
```

Format

A data frame with 20 observations on the following 6 variables.

`salaryP` staff salaries per pupil

`fatherWc` percent of white-collar fathers

`sstatus` socioeconomic status composite deviation: means for family size, family intactness, father's education, mother's education, and home items

`teacherSc` mean teacher's verbal test score

`motherLev` mean mother's educational level, one unit is equal to two school years

`Y` verbal mean test score (y, all sixth graders)

Author(s)

Valentin Todorov

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection* Wiley, p.79, table 2.

Examples

```
data(coleman)
pairs(coleman)
summary(lm.coleman <- lm(Y ~ . , data = coleman))
summary(lts.coleman <- ltsReg(Y ~ . , data = coleman))

coleman.x <- data.matrix(coleman[, 1:6])
(Cc <- covMcd(coleman.x))
```

`condroz`*Condroz Data*

Description

Dataset with pH-value and Calcium content in soil samples, collected in different communities of the Condroz region in Belgium. The data pertain to a subset of 428 samples with a pH-value between 7.0 and 7.5.

Usage

```
data(condroz)
```

Format

A data frame with 428 observations on the following 2 variables.

Ca Calcium content of the soil sample

pH pH value of the soil sample

Details

For more information on the dataset, cf. Goegebeur et al. (2005).

Source

Hubert and Vandervieren (2006), p. 10. This dataset is also studied in Vandewalle et al. (2004).

References

Goegebeur, Y., Planchon, V., Beirlant, J., Oger, R. (2005). Quality Assesment of Pedochemical Data Using Extreme Value Methodology, *Journal of Applied Science*, 5, p. 1092-1102.

Hubert, M. and Vandervieren, E. (2006). An Adjusted Boxplot for Skewed Distributions, Technical Report TR-06-11, KULeuven, Section of Statistics, Leuven. <http://wis.kuleuven.be/stat/robust/Papers/TR0611.pdf>

Vandewalle, B., Beirlant, J., Hubert, M. (2004). A robust estimator of the tail index based on an exponential regression model, in Hubert, M., Pison G., Struyf, A. and S. Van Aelst, ed., *Theory and Applications of Recent Robust Methods*, Birkhäuser, Basel, p. 367-376.

Examples

```
adjbox(condroz$Ca)
```

covMcd

*Robust Location and Scatter Estimation via MCD***Description**

Compute a robust multivariate location and scale estimate with a high breakdown point, using the 'Fast MCD' (Minimum Covariance Determinant) estimator.

Usage

```
covMcd(x, cor = FALSE, alpha = 1/2, nsamp = 500, nmini = 300, seed = NULL,
       trace = FALSE, use.correction = TRUE, control = rrcov.control())
```

Arguments

x	a matrix or data frame.
cor	should the returned result include a correlation matrix? Default is cor = FALSE.
alpha	numeric parameter controlling the size of the subsets over which the determinant is minimized, i.e., alpha*n observations are used for computing the determinant. Allowed values are between 0.5 and 1 and the default is 0.5.
nsamp	number of subsets used for initial estimates or "best" or "exact". Default is nsamp = 500. For nsamp = "best" exhaustive enumeration is done, as long as the number of trials does not exceed 100'000 (= nLarge). For "exact", exhaustive enumeration will be attempted however many samples are needed. In this case a warning message may be displayed saying that the computation can take a very long time.
nmini	for large n, the algorithm splits the data into maximally $k_{rep} = 5$ subsets of size nmini. The original algorithm had nmini = 300 hard coded.
seed	initial seed for random generator, see rrcov.control .
trace	logical (or integer) indicating if intermediate results should be printed; defaults to FALSE; values ≥ 2 also produce print from the internal (Fortran) code.
use.correction	whether to use finite sample correction factors; defaults to TRUE.
control	a list with estimation options - this includes those above provided in the function specification, see rrcov.control for the defaults. If control 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.

Details

The minimum covariance determinant estimator of location and scatter implemented in covMcd() is similar to R function [cov.mcd\(\)](#) in **MASS**. The MCD method looks for the $h(> n/2)$ ($h = h(\alpha, n, p) = h.alpha.n(\alpha, n, p)$) observations (out of n) whose classical covariance matrix has the lowest possible determinant.

The raw MCD estimate of location is then the average of these h points, whereas the raw MCD estimate of scatter is their covariance matrix, multiplied by a consistency factor and a finite sample correction factor (to make it consistent at the normal model and unbiased at small samples).

The implementation of `covMcd` uses the Fast MCD algorithm of Rousseeuw and Van Driessen (1999) to approximate the minimum covariance determinant estimator.

Both rescaling factors (consistency and finite sample) are returned also in the vector `raw.cnp2` of length 2. Based on these raw MCD estimates, a reweighting step is performed which increases the finite-sample efficiency considerably - see Pison et al. (2002). The rescaling factors for the reweighted estimates are returned in the vector `cnp2` of length 2. Details for the computation of the finite sample correction factors can be found in Pison et al. (2002).

The finite sample corrections can be suppressed by setting `use.correction = FALSE`.

Value

An object of class "mcd" which is basically a `list` with components

<code>center</code>	the final estimate of location.
<code>cov</code>	the final estimate of scatter.
<code>cor</code>	the (final) estimate of the correlation matrix (only if <code>cor = TRUE</code>).
<code>crit</code>	the value of the criterion, i.e. the determinant.
<code>best</code>	the best subset found and used for computing the raw estimates, with <code>length(best) == quan = h.alpha.n(alpha, n, p)</code> .
<code>mah</code>	mahalanobis distances of the observations using the final estimate of the location and scatter.
<code>mcd.wt</code>	weights of the observations using the final estimate of the location and scatter.
<code>cnp2</code>	a vector of length two containing the consistency correction factor and the finite sample correction factor of the final estimate of the covariance matrix.
<code>raw.center</code>	the raw (not reweighted) estimate of location.
<code>raw.cov</code>	the raw (not reweighted) estimate of scatter.
<code>raw.mah</code>	mahalanobis distances of the observations based on the raw estimate of the location and scatter.
<code>raw.weights</code>	weights of the observations based on the raw estimate of the location and scatter.
<code>raw.cnp2</code>	a vector of length two containing the consistency correction factor and the finite sample correction factor of the raw estimate of the covariance matrix.
<code>X</code>	the input data as numeric matrix, without NAs.
<code>n.obs</code>	total number of observations.
<code>alpha</code>	the size of the subsets over which the determinant is minimized (the default is $(n + p + 1)/2$).
<code>quan</code>	the number of observations, h , on which the MCD is based. If <code>quan</code> equals <code>n.obs</code> , the MCD is the classical covariance matrix.
<code>method</code>	character string naming the method (Minimum Covariance Determinant).
<code>call</code>	the call used (see <code>match.call</code>).

Author(s)

Valentin Todorov <valentin.todorov@chello.at>, based on work written for S-plus by Peter Rousseeuw and Katrien van Driessen from University of Antwerp.

References

- P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*. Wiley.
- P. J. Rousseeuw and K. van Driessen (1999) A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **41**, 212–223.
- Pison, G., Van Aelst, S., and Willems, G. (2002), Small Sample Corrections for LTS and MCD, *Metrika*, **55**, 111-123.

See Also

`cov.mcd` from package **MASS**; `covOGK` as cheaper alternative for larger dimensions.

Examples

```
data(hbk)
hbk.x <- data.matrix(hbk[, 1:3])
covMcd(hbk.x)

## the following three statements are equivalent
c1 <- covMcd(hbk.x, alpha = 0.75)
c2 <- covMcd(hbk.x, control = rrcov.control(alpha = 0.75))
## direct specification overrides control one:
c3 <- covMcd(hbk.x, alpha = 0.75,
             control = rrcov.control(alpha=0.95))

c1
```

covOGK

Orthogonalized Gnanadesikan-Kettenring (OGK) Covariance Matrix Estimation

Description

Computes the orthogonalized pairwise covariance matrix estimate described in in Maronna and Zamar (2002). The pairwise proposal goes back to Gnanadesikan and Kettenring (1972).

Usage

```
covOGK(X, n.iter = 2, sigmamu, rcov = covGK, weight.fn = hard.rejection,
       keep.data = FALSE, ...)
```

```
covGK(x, y, scalefn = scaleTau2, ...)
s_mad(x, mu.too = FALSE, na.rm = FALSE)
s_IQR(x, mu.too = FALSE, na.rm = FALSE)
```

Arguments

<code>x</code>	data in something that can be coerced into a numeric matrix.
<code>n.iter</code>	number of orthogonalization iterations. Usually 1 or 2; values greater than 2 are unlikely to have any significant effect on the estimate (other than increasing the computing time).
<code>sigmamu, scalefn</code>	a function that computes univariate robust location and scale estimates. By default it should return a single numeric value containing the robust scale (standard deviation) estimate. When <code>mu.too</code> is true, <code>sigmamu()</code> should return a numeric vector of length 2 containing robust location and scale estimates. See scaleTau2 , s_Qn , s_Sn , s_mad or s_IQR for examples to be used as <code>sigmamu</code> argument.
<code>rcov</code>	function that computes a robust covariance estimate between two vectors. The default, Gnanadesikan-Kettenring's <code>covGK</code> , is simply $(s^2(X + Y) - s^2(X - Y))/4$ where $s()$ is the scale estimate <code>sigmamu()</code> .
<code>weight.fn</code>	a function of the robust distances and the number of variables p to compute the weights used in the reweighting step.
<code>keep.data</code>	logical indicating if the (untransformed) data matrix <code>X</code> should be kept as part of the result.
<code>...</code>	additional arguments; for <code>covOGK</code> to be passed to <code>sigmamu()</code> and <code>weight.fn()</code> ; for <code>covGK</code> passed to <code>scalefn</code> .
<code>x, y</code>	numeric vectors of the same length, the covariance of which is sought in <code>covGK</code> (or the scale, in <code>s_mad</code> or <code>s_IQR</code>).
<code>mu.too</code>	logical indicating if both location and scale should be returned or just the scale (when <code>mu.too=FALSE</code> as by default).
<code>na.rm</code>	if TRUE then NA values are stripped from <code>x</code> before computation takes place.

Details

Typical default values for the *function* arguments `sigmamu`, `rcov`, and `weight.fn`, are available as well, see the *Examples* below, **but** their names and calling sequences are still subject to discussion and may be changed in the future.

The current default, `weight.fn = hard.rejection` corresponds to the proposition in the literature, but Martin Maechler strongly believes that the hard threshold currently in use is too arbitrary, and further that *soft* thresholding should be used instead, anyway.

Value

`covOGK()` currently returns a list with components

<code>center</code>	robust location: numeric vector of length p .
<code>cov</code>	robust covariance matrix estimate: $p \times p$ matrix.
<code>wcenter, wcov</code>	re-weighted versions of <code>center</code> and <code>cov</code> .
<code>weights</code>	the robustness weights used.
<code>distances</code>	the mahalanobis distances computed using <code>center</code> and <code>cov</code> .

.....

but note that this might be radically changed to returning an S4 classed object!

covGK() is a trivial 1-line function returning the covariance estimate

$$\hat{c}(x, y) = (\hat{\sigma}(x + y)^2 - \hat{\sigma}(x - y)^2) / 4,$$

where $\hat{\sigma}(u)$ is the scale estimate of u specified by scalefn.

s_mad(), and s_IQR() return the scale estimates [mad](#) or [IQR](#) respectively, where the s_* functions return a length-2 vector (mu, sig) when mu.too = TRUE, see also [scaleTau2](#).

Author(s)

Kjell Konis <konis@stats.ox.ac.uk>, with modifications by Martin Maechler.

References

Maronna, R.A. and Zamar, R.H. (2002) Robust estimates of location and dispersion of high-dimensional datasets; *Technometrics* **44**(4), 307–317.

Gnanadesikan, R. and John R. Kettenring (1972) Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics* **28**, 81–124.

See Also

[scaleTau2](#), [covMcd](#), [cov.rob](#).

Examples

```
data(hbk)
hbk.x <- data.matrix(hbk[, 1:3])

c01 <- covOGK(hbk.x, sigmamu = scaleTau2)
c02 <- covOGK(hbk.x, sigmamu = s_Qn)
c03 <- covOGK(hbk.x, sigmamu = s_Sn)
c04 <- covOGK(hbk.x, sigmamu = s_mad)
c05 <- covOGK(hbk.x, sigmamu = s_IQR)

data(toxicity)
c01tox <- covOGK(toxicity, sigmamu = scaleTau2)
c02tox <- covOGK(toxicity, sigmamu = s_Qn)

## nice formatting of correlation matrices:
as.dist(round(cov2cor(c01tox$cov), 2))
as.dist(round(cov2cor(c02tox$cov), 2))

## "graphical"
symnum(cov2cor(c01tox$cov))
symnum(cov2cor(c02tox$cov), legend=FALSE)
```

CrohnD

Crohn's Disease Adverse Events Data

Description

Data set issued from a study of the adverse events of a drug on 117 patients affected by Crohn's disease (a chronic inflammatory disease of the intestines).

Usage

```
data(CrohnD)
```

Format

A data frame with 117 observations on the following 9 variables.

ID the numeric patient IDs

nrAdvE the number of adverse events

BMI Body MASS Index, i.e., $weight[kg]/(height[m])^2$.

height in cm

country a factor with levels 0 and 1

sex the person's gender, a binary factor with levels M F

age in years, a numeric vector

weight in kilograms, a numeric vector

treat how CD was treated: a factor with levels 0, 1 and 2, meaning placebo, drug 1 and drug 2.

Source

form the authors of the reference, with permission by the original data collecting agency.

References

Serigne N. Lô and Elvezio Ronchetti (2006). Robust Second Order Accurate Inference for Generalized Linear Models. Technical report, University of Geneva, Switzerland.

Examples

```
data(CrohnD)
str(CrohnD)
with(CrohnD, ftable(table(sex, country, treat)))
```

 cushny

Cushny and Peebles Prolongation of Sleep Data

Description

The original data set was bivariate and recorded for ten subjects the prolongation of sleep caused by two different drugs. These data were used by Student as the first illustration of the paired t-test which only needs the *differences* of the two measurements. These differences are the values of cushny.

Usage

```
data(cushny)
```

Format

numeric vector, sorted increasingly:
0 0.8 1 1.2 1.3 1.3 1.4 1.8 2.4 4.6

Source

Cushny, A.R. and Peebles, A.R. (1905) The action of optical isomers. II. Hyoscines. *J. Physiol.* **32**, 501–510.

These data were used by Student(1908) as the first illustration of the paired t-test, see also [sleep](#); then cited by Fisher (1925) and thereafter copied in numerous books as an example of a normally distributed sample, see, e.g., Anderson (1958).

References

Student (1908) The probable error of a mean. *Biometrika* **6**, 1–25.

Fisher, R.A. (1925) *Statistical Methods for Research Workers*; Oliver & Boyd, Edinburgh.

Anderson, T.W. (1958) *An Introduction to Multivariate Statistical Analysis*; Wiley, N.Y.

Hampel, F., Ronchetti, E., Rousseeuw, P. and Stahel, W. (1986) *Robust Statistics: The Approach Based on Influence Functions*; Wiley, N.Y.

Examples

```
data(cushny)

plot(cushny, rep(0, 10), pch = 3, cex = 3,
      ylab = "", yaxt = "n")
plot(jitter(cushny), rep(0, 10), pch = 3, cex = 2,
      main = "'cushny' data (n= 10)", ylab = "", yaxt = "n")
abline(h=0, col="gray", lty=3)
myPt <- function(m, lwd = 2, ..., e = 1.5*par("cxy")[2])
  segments(m, +e, m, -e, lwd = lwd, ...)
myPt( mean(cushny), col = "pink3")
```

```

myPt(median(cushny), col = "light blue")
legend("topright", c("mean", "median"), lwd = 2,
      col = c("pink3", "light blue"), inset = .01)

## The 'sleep' data from the standard 'datasets' package:
d.sleep <- local({ gr <- with(datasets::sleep, split(extra, group))
                  gr[[2]] - gr[[1]] })
stopifnot(all.equal(cushny,
                    sort(d.sleep), tol=1e-15))

```

 delivery

Delivery Time Data

Description

Delivery Time Data, from Montgomery and Peck (1982). The aim is to explain the time required to service a vending machine (Y) by means of the number of products stocked (X1) and the distance walked by the route driver (X2).

Usage

```
data(delivery)
```

Format

A data frame with 25 observations on the following 3 variables.

n.prod Number of Products

distance Distance

delTime Delivery time

Source

Montgomery and Peck (1982, p.116)

References

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, page 155, table 23.

Examples

```

data(delivery)
summary(lm.deli <- lm(delTime ~ ., data = delivery))

delivery.x <- as.matrix(delivery[, 1:2])
c_deli <- covMcd(delivery.x)
c_deli

```

education

Education Expenditure Data

Description

Education Expenditure Data, from Chatterjee and Price (1977, p.108). This data set, representing the education expenditure variables in the 50 US states, providing an interesting example of heteroscedacity.

Usage

```
data(education)
```

Format

A data frame with 50 observations on the following 6 variables.

State State

Region Region (1=Northeastern, 2=North central, 3=Southern, 4=Western)

X1 Number of residents per thousand residing in urban areas in 1970

X2 Per capita personal income in 1973

X3 Number of residents per thousand under 18 years of age in 1974

Y Per capita expenditure on public education in a state, projected for 1975

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.110, table 16.

Examples

```
data(education)
```

```
education.x <- data.matrix(education[, 3:5])
```

```
summary(lm.education <- lm(Y ~ Region + X1+X2+X3, data=education))
```

epilepsy

Epilepsy Attacks Data Set

Description

Data from a clinical trial of 59 patients with epilepsy (Breslow, 1996) in order to illustrate diagnostic techniques in Poisson regression.

Usage

```
data(epilepsy)
```

Format

A data frame with 59 observations on the following 11 variables.

ID Patient identification number

Y1 Number of epilepsy attacks patients have during the first follow-up period

Y2 Number of epilepsy attacks patients have during the second follow-up period

Y3 Number of epilepsy attacks patients have during the third follow-up period

Y4 Number of epilepsy attacks patients have during the fourth follow-up period

Base Number of epileptic attacks recorded during 8 week period prior to randomization

Age Age of the patients

Trt a factor with levels placebo progabide indicating whether the anti-epilepsy drug Progabide has been applied or not

Ysum Total number of epilepsy attacks patients have during the four follow-up periods

Age10 Age of the patients divided by 10

Base4 Variable Base divided by 4

Details

Thall and Vail reported data from a clinical trial of 59 patients with epilepsy, 31 of whom were randomized to receive the anti-epilepsy drug Progabide and 28 of whom received a placebo. Baseline data consisted of the patient's age and the number of epileptic seizures recorded during 8 week period prior to randomization. The response consisted of counts of seizures occurring during the four consecutive follow-up periods of two weeks each.

Source

Thall, P.F. and Vail S.C. (1990) Some covariance models for longitudinal count data with overdispersion. *Biometrics* **46**, 657–671.

References

- Diggle, P.J., Liang, K.Y., and Zeger, S.L. (1994) *Analysis of Longitudinal Data*; Clarendon Press.
- Breslow N. E. (1996) Generalized linear models: Checking assumptions and strengthening conclusions. *Statistica Applicata* **8**, 23–41.

Examples

```
data(epilepsy)
str(epilepsy)
pairs(epilepsy[,c("Ysum", "Base4", "Trt", "Age10")])

Efit1 <- glm(Ysum ~ Age10 + Base4*Trt, family=poisson, data=epilepsy)
summary(Efit1)

## Robust Fit : %>>>>> FIXME <<<<<
```

exAM

Example Data of Antille and May - for Simple Regression

Description

This is an artificial data set, cleverly constructed and used by Antille and May to demonstrate ‘problems’ with LMS and LTS.

Usage

```
data(exAM)
```

Format

A data frame with 12 observations on 2 variables, x and y.

Details

Because the points are not in general position, both LMS and LTS typically *fail*; however, e.g., `rlm(*, method="MM")` “works”.

Source

Antille, G. and El May, H. (1992) The use of slices in the LMS and the method of density slices: Foundation and comparison. In Yadolah Dodge and Joe Whittaker, editors, *COMPSTAT: Proc. 10th Symp. Computat. Statist., Neuchatel*, **1**, 441–445; Physica-Verlag.

Examples

```
data(exAM)
plot(exAM)
summary(ls <- lm(y ~ x, data=exAM))
abline(ls)
```

functionX-class

Class "functionX" of Psi-like Vectorized Functions

Description

The class "functionX" of vectorized functions of one argument x and typically further tuning parameters.

Objects from the Class

Objects can be created by calls of the form `new("functionX", ...)`.

Slots

.Data: Directly extends class "function".

Extends

Class "function", from data part. Class "OptionalFunction", by class "function". Class "PossibleMethod", by class "function".

Methods

No methods defined with class "functionX" in the signature.

Author(s)

Martin Maechler

See Also

[functionXal-class](#) for *functionals* of "functionX"; [psiFunc](#), and [psi_func-class](#) which has several functionX slots.

functionXal-class *Class "functionXal" of Functionals (of Psi-like functions)*

Description

The class "functionXal" is a class of functionals (typically integrals) typically of [functionX-class](#) functions.

Since the functionX functions typically also depend on tuning parameters, objects of this class ("functionXal") are functions of these tuning parameters.

Slots

.Data: Directly extends class "function".

Extends

Class "function", from data part. Class "OptionalFunction", by class "function". Class "PossibleMethod", by class "function".

See Also

[functionX-class](#); [psiFunc](#), and [psi_func-class](#) which has several functionXal slots.

glmrob *Robust Fitting of Generalized Linear Models*

Description

glmrob is used to fit generalized linear models by robust methods. The models are specified by giving a symbolic description of the linear predictor and a description of the error distribution. Currently, robust methods are implemented for `family = binomial`, `= poisson`, `= Gamma` and `= gaussian`.

Usage

```
glmrob(formula, family, data, weights, subset, na.action,
       start = NULL, offset, method = "Mqle",
       weights.on.x = c("none", "hat", "robCov", "covMcd"), control = NULL,
       model = TRUE, x = FALSE, y = TRUE, contrasts = NULL, trace = FALSE, ...)
```

Arguments

formula	a formula , i.e., a symbolic description of the model to be fit (cf. glm or lm).
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) As mentioned, currently this must be <code>binomial</code> or <code>poisson</code> .
data	an optional data frame containing the variables in the model. If not found in <code>data</code> , the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>glmrob</code> is called.
weights	an optional vector of weights to be used in the fitting process.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the <code>na.action</code> setting in options . The “factory-fresh” default is <code>na.omit</code> .
start	starting values for the parameters in the linear predictor.
offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting.
method	a character string specifying the robust fitting method. The details of method specification are given below.
weights.on.x	character string (can be abbreviated) specifying how points (potential outliers) in <i>x</i> -space are downweighted. If “hat”, weights on the design of the form $\sqrt{1-h_{ii}}$ are used, where h_{ii} are the diagonal elements of the hat matrix. If “robCov”, weights based on the robust Mahalanobis distance of the design matrix (intercept excluded) are used where the covariance matrix and the centre is estimated by <code>cov.rob</code> from the package MASS . Similarly, if “covMcd”, robust weights are computed using <code>covMcd</code> . The default is “none”.
control	a list of parameters for controlling the fitting process. See the documentation for glmrobMqle.control for details.
model	a logical value indicating whether <i>model frame</i> should be included as a component of the returned value.
x, y	logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value.
contrasts	an optional list. See the <code>contrasts.arg</code> of <code>model.matrix.default</code> .
trace	logical (or integer) indicating if intermediate results should be printed; defaults to <code>FALSE</code> .
...	arguments passed to glmrobMqle.control when <code>control</code> is <code>NULL</code> (as per default).

Details

`method="Mqle"` fits a generalized linear model using Mallows or Huber type robust estimators, as described in Cantoni and Ronchetti (2001) and Cantoni and Ronchetti (2006). In contrast to the

implementation described in Cantoni (2004), the pure influence algorithm is implemented. Currently no other method is implemented.

`weights.on.x = "robCov"` makes sense if all explanatory variables are continuous.

Value

`glmrob` returns an object of class `"glmrob"` and is also inheriting from `glm`. The `summary` method, see `summary.glmrob`, can be used to obtain or print a summary of the results. The generic accessor functions `coefficients`, `effects`, `fitted.values` and `residuals` (see `residuals.glmrob`) can be used to extract various useful features of the value returned by `glmrob()`. An object of class `"glmrob"` is a list with at least the following components:

<code>coefficients</code>	a named vector of coefficients
<code>residuals</code>	the <i>working</i> residuals, that is the (robustly “huberized”) residuals in the final iteration of the IWLS fit.
<code>fitted.values</code>	the fitted mean values, obtained by transforming the linear predictors by the inverse of the link function.
<code>w.r</code>	robustness weights for each observations; i.e., $\text{residuals} \times \text{w.r}$ equals the psi-function of the Preason’s residuals.
<code>w.x</code>	weights used to down-weight observations based on the position of the observation in the design space.
<code>dispersion</code>	robust estimation of dispersion paramter if appropriate
<code>cov</code>	the estimated asymptotic covariance matrix of the estimated coefficients.
<code>tcc</code>	the tuning constant c in Huber’s psi-function.
<code>family</code>	the <code>family</code> object used.
<code>linear.predictors</code>	the linear fit on link scale.
<code>deviance</code>	NULL; Exists because of compaitility reasons.
<code>iter</code>	the number of iterations used by the influence algorithm.
<code>converged</code>	logical. Was the IWLS algorithm judged to have converged?
<code>call</code>	the matched call.
<code>formula</code>	the formula supplied.
<code>terms</code>	the <code>terms</code> object used.
<code>data</code>	the <code>data</code> argument.
<code>offset</code>	the offset vector used.
<code>control</code>	the value of the <code>control</code> argument used.
<code>method</code>	the name of the robust fitter function used.
<code>contrasts</code>	(where relevant) the contrasts used.
<code>xlevels</code>	(where relevant) a record of the levels of the factors used in fitting.

Author(s)

Andreas Ruckstuhl

References

- E. Cantoni and E. Ronchetti (2001) Robust Inference for Generalized Linear Models. *JASA* **96** (455), 1022–1030.
- E. Cantoni (2004) Analysis of Robust Quasi-deviances for Generalized Linear Models. *Journal of Statistical Software*, **10**, <http://www.jstatsoft.org/v10/i04>
- E. Cantoni and E. Ronchetti (2006) A robust approach for skewed and heavy-tailed outcomes in the analysis of health care expenditures. *Journal of Health Economics* **25**, 198–213.
- S. Heritier, E. Cantoni, S. Copt, M.-P. Victoria-Feser (2009) *Robust Methods in Biostatistics*. Wiley Series in Probability and Statistics.

See Also

[predict.glmrob](#) for prediction; [glmrobMqle.control](#)

Examples

```
## Binomial response -----
data(carrots)

Cfit1 <- glm(cbind(success, total-success) ~ logdose + block,
            data = carrots, family = binomial)
summary(Cfit1)

Rfit1 <- glmrob(cbind(success, total-success) ~ logdose + block,
              family = binomial, data = carrots, method = "Mqle",
              control = glmrobMqle.control(tcc=1.2))
summary(Rfit1)

Rfit2 <- glmrob(success/total ~ logdose + block, weights = total,
              family = binomial, data = carrots, method = "Mqle",
              control = glmrobMqle.control(tcc=1.2))
coef(Rfit2) ## The same as Rfit1

## Binary response -----
data(vaso)

Vfit1 <- glm(Y ~ log(Volume) + log(Rate), family=binomial, data=vaso)
coef(Vfit1)

Vfit2 <- glmrob(Y ~ log(Volume) + log(Rate), family=binomial, data=vaso,
              method="Mqle", control = glmrobMqle.control(tcc=3.5))
## Note the problems with tcc <= 3 %% FIXME algorithm ???
coef(Vfit2) # c = 3.5 ==> not much different from classical

## Poisson response -----
data(epilepsy)
```

```

Efit1 <- glm(Ysum ~ Age10 + Base4*Trt, family=poisson, data=epilepsy)
summary(Efit1)

Efit2 <- glmrob(Ysum ~ Age10 + Base4*Trt, family = poisson,
               data = epilepsy, method= "Mqle",
               control = glmrobMqle.control(tcc= 1.2))
summary(Efit2)

## 'x' weighting:
(Efit3 <- glmrob(Ysum ~ Age10 + Base4*Trt, family = poisson,
               data = epilepsy, method= "Mqle", weights.on.x = "hat",
               control = glmrobMqle.control(tcc= 1.2)))

try( # gives singular cov matrix: 'Trt' is binary factor -->
    # affine equivariance and subsampling are problematic
Efit4 <- glmrob(Ysum ~ Age10 + Base4*Trt, family = poisson,
               data = epilepsy, method= "Mqle", weights.on.x = "covMcd",
               control = glmrobMqle.control(tcc=1.2, maxit=100))
)

### ----- Gamma family -- data from example(glm) ---

clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))
summary(cl <- glm (lot1 ~ log(u), data=clotting, family=Gamma))
summary(ro <- glmrob(lot1 ~ log(u), data=clotting, family=Gamma))

clotM5.high <- within(clotting, { lot1[5] <- 60 })
op <- par(mfrow=2:1, mgp = c(1.6, 0.8, 0), mar = c(3,3:1))
plot( lot1 ~ log(u), data=clotM5.high)
plot(1/lot1 ~ log(u), data=clotM5.high)
par(op)
## Obviously, there the first observation is an outlier with respect to both
## representations!

cl5.high <- glm (lot1 ~ log(u), data=clotM5.high, family=Gamma)
ro5.high <- glmrob(lot1 ~ log(u), data=clotM5.high, family=Gamma)
with(ro5.high, cbind(w.x, w.r))## the 5th obs. is downweighted heavily!

plot(1/lot1 ~ log(u), data=clotM5.high)
abline(cl5.high, lty=2, col="red")
abline(ro5.high, lwd=2, col="blue") ## result is ok (but not "perfect")

```

glmrobMqle.control *Auxiliary for Controlling Robust GLM Fitting by the Method "Mqle"*

Description

Auxiliary function as user interface for [glmrob](#) fitting when method "Mqle" is applied. Typically only used when calling [glmrob](#).

Usage

```
glmrobMqle.control(acc = 1e-04, test.acc = "coef", maxit = 50, tcc = 1.345)
```

Arguments

acc	positive convergence tolerance; the iterations converge when ???
test.acc	Only "coef" is currently implemented
maxit	integer giving the maximum number of iterations.
tcc	tuning constant c for Huber's psi-function

Value

A list with the arguments as components.

Author(s)

Andreas Ruckstuhl

See Also

[glmrob](#)

h.alpha.n

Compute h , the subsample size for MCD and LTS

Description

Compute $h(\alpha)$ which is the size of the subsamples to be used for MCD and LTS. Given $\alpha = \alpha$, n and p , h is an integer, $h \approx \alpha n$, where the exact formula also depends on p .

For $\alpha = 1/2$, $h == \text{floor}((n+p+1)/2)$; for the general case, it's simply $n2 <- (n+p+1) \% \% 2$; $\text{floor}(2*n2 - n + 2*(n-n2)*$

Usage

```
h.alpha.n(alpha, n, p)
```

Arguments

alpha	fraction, numeric (vector) in [0.5, 1], see, e.g., covMcd .
n	integer (valued vector), the sample size.
p	integer (valued vector), the dimension.

Value

numeric vector of $h(\alpha, n, p)$; when any of the arguments of length greater than one, the usual R arithmetic (recycling) rules are used.

See Also

[covMcd](#) and [ltsReg](#) which are *defined* by $h = h(\alpha, n, p)$ and hence both use `h.alpha.n`.

Examples

```
n <- c(10:20, 50, 100)
p <- 5
## show the simple "alpha = 1/2" case:
cbind(n=n, h= h.alpha.n(1/2, n, p), n2p = floor((n+p+1)/2))

## alpha = 3/4 is recommended by some authors :
n <- c(15, 20, 25, 30, 50, 100)
cbind(n=n, h= h.alpha.n(3/4, n, p = 6))
```

hbk

Hawkins, Bradu, Kass's Artificial Data

Description

Artificial Data Set generated by Hawkins, Bradu, and Kass (1984). The data set consists of 75 observations in four dimensions (one response and three explanatory variables). It provides a good example of the masking effect. The first 14 observations are outliers, created in two groups: 1–10 and 11–14. Only observations 12, 13 and 14 appear as outliers when using classical methods, but can be easily unmasked using robust distances computed by, e.g., MCD - `covMcd()`.

Usage

```
data(hbk)
```

Format

A data frame with 75 observations on 4 variables, where the last variable is the dependent one.

X1 x[,1]

X2 x[,2]

X3 x[,3]

Y y

Note

This data set is also available in package **wle** as `artificial`.

Source

Hawkins, D.M., Bradu, D., and Kass, G.V. (1984) Location of several outliers in multiple regression data using elemental sets. *Technometrics* **26**, 197–208.

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.94.

Examples

```
data(hbk)
plot(hbk)
summary(lm.hbk <- lm(Y ~ ., data = hbk))

hbk.x <- data.matrix(hbk[, 1:3])
(cHBK <- covMcd(hbk.x))
```

heart

Heart Catherization Data

Description

This data set was analyzed by Weisberg (1980) and Chambers et al. (1983). A catheter is passed into a major vein or artery at the femoral region and moved into the heart. The proper length of the introduced catheter has to be guessed by the physician. The aim of the data set is to describe the relation between the catheter length and the patient's height (X1) and weight (X2).

This data sets is used to demonstrate the effects caused by collinearity. The correlation between height and weight is so high that either variable almost completely determines the other.

Usage

```
data(heart)
```

Format

A data frame with 12 observations on the following 3 variables.

height Patient's height in inches

weight Patient's weights in pounds

clength Y: Catheter Length (in centimeters)

Note

There are other heart datasets in other R packages, notably **survival**, hence considering using `package = "robustbase"`, see examples.

Source

Weisberg (1980)

Chambers et al. (1983)

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.103, table 13.

Examples

```
data(heart, package="robustbase")
heart.x <- data.matrix(heart[, 1:2]) # the X-variables
plot(heart.x)
covMcd(heart.x)
summary(lm.heart <- lm(clength ~ . , data = heart))
summary(lts.heart <- ltsReg(clength ~ . , data = heart))
```

`huberM`*Safe (generalized) Huber M-Estimator of Location*

Description

(Generalized) Huber M-estimator of location with MAD scale, being sensible also when the scale is zero where `huber()` returns an error.

Usage

```
huberM(x, k = 1.5, weights = NULL, tol = 1e-06,  
       mu = if(is.null(weights)) median(x) else wgt.himedian(x, weights),  
       s = if(is.null(weights)) mad(x, center=mu)  
       else wgt.himedian(abs(x - mu), weights),  
       warn0scale = getOption("verbose"))
```

Arguments

<code>x</code>	numeric vector.
<code>k</code>	positive factor; the algorithm winsorizes at <code>k</code> standard deviations.
<code>weights</code>	numeric vector of non-negative weights of same length as <code>x</code> , or <code>NULL</code> .
<code>tol</code>	convergence tolerance.
<code>mu</code>	initial location estimator.
<code>s</code>	scale estimator held constant through the iterations.
<code>warn0scale</code>	logical; if true, and <code>s</code> is 0 and <code>length(x) > 1</code> , this will be warned about.

Details

Note that currently, when non-`NULL` `weights` are specified, the default for initial location `mu` and scale `s` is `wgt.himedian`, where strictly speaking a weighted “non-hi” median should be used for consistency. Since `s` is not updated, the results slightly differ, see the examples below.

Value

list of location and scale parameters, and number of iterations used.

<code>mu</code>	location estimate
<code>s</code>	the <code>s</code> argument, typically the <code>mad</code> .
<code>it</code>	the number of “Huber iterations” used.

Author(s)

Martin Maechler, building on the MASS code mentioned.

References

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

See Also

[hubers](#) (and [huber](#)) in package **MASS**; [mad](#).

Examples

```
huberM(c(1:9, 1000))
mad (c(1:9, 1000))
mad (rep(9, 100))
huberM(rep(9, 100))

## When you have "binned" aka replicated observations:
set.seed(7)
x <- c(round(rnorm(1000),1), round(rnorm(50, m=10, sd = 10)))
t.x <- table(x) # -> unique values and multiplicities
x.uniq <- as.numeric(names(t.x)) ## == sort(unique(x))
x.mult <- unname(t.x)
str(Hx <- huberM(x.uniq, weights = x.mult), digits = 7)
str(Hx. <- huberM(x, s = Hx$s), digits = 7) ## should be ~ Hx
stopifnot(all.equal(Hx, Hx.))
str(Hx2 <- huberM(x), digits = 7)## somewhat different, since 's' differs
```

kootenay

(Modified) Waterflow Measurements of Kootenay River in Libby and Newgate

Description

The original data set is the waterflow in January of the Kootenay river, measured at two locations, namely, Libby (Montana) and Newgate (British Columbia) for 13 consecutive years, 1931–1943.

The data set is of mostly interest because it has been used as example in innumerable didactical situations about robust regression. To this end, one number (in observation 4) has been modified from the original data from originally 44.9 to 15.7 (here).

Usage

```
data(kootenay)
```

Format

A data frame with 13 observations on the following 2 variables.

Libby a numeric vector

Newgate a numeric vector

Details

Other modified versions of the data sets are also used in different places, see the examples below.

Source

Original Data, p.58f of Ezekiel and Fox (1959), *Methods of Correlation and Regression Analysis*. Wiley, N.Y.

References

Hampel, F., Ronchetti, E., Rousseeuw, P. and Stahel, W. (1986) *Robust Statistics: The Approach Based on Influence Functions*; Wiley, N.Y.

Rousseeuw, P. J. and Leroy, A. M. (1987) *Robust Regression & Outlier Detection*, Wiley, N. Y.

Examples

```
data(kootenay)
plot(kootenay, main = "'kootenay' data")
points(kootenay[4,], col = 2, cex =2, pch = 3)

abline(lm (Newgate ~ Libby, data = kootenay), col = "pink")
abline(lmrob(Newgate ~ Libby, data = kootenay), col = "blue")

## The original version of Ezekiel & Fox:
kootenay0 <- kootenay
kootenay0[4, "Newgate"] <- 44.9
plot(kootenay0, main = "'kootenay0': the original data")
abline(lm (Newgate ~ Libby, data = kootenay0), col = "pink")
abline(lmrob(Newgate ~ Libby, data = kootenay0), col = "blue")

## The version with "milder" outlier -- Hampel et al., p.310
kootenay2 <- kootenay0
kootenay2[4, "Libby"] <- 20.0 # instead of 77.6
plot(kootenay2, main = "The 'kootenay2' data",
     xlim = range(kootenay[, "Libby"]))
points(kootenay2[4,], col = 2, cex =2, pch = 3)
abline(lm (Newgate ~ Libby, data = kootenay2), col = "pink")
abline(lmrob(Newgate ~ Libby, data = kootenay2), col = "blue")
```

lactic

Lactic Acid Concentration Measurement Data

Description

Data on the Calibration of an Instrument that Measures Lactic Acid Concentration in Blood, from Afifi and Azen (1979) - comparing the true concentration X with the measured value Y.

Usage

```
data(lactic)
```

Format

A data frame with 20 observations on the following 2 variables.

X True Concentration

Y Instrument

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.62, table 10.

Examples

```
data(lactic)
summary(lm.lactic <- lm(Y ~., data=lactic))
```

 lmrob

MM-type Estimators for Linear Regression

Description

Computes fast MM-type estimators for linear (regression) models.

Usage

```
lmrob(formula, data, subset, weights, na.action, method = "MM",
       model = TRUE, x = !control$compute.rd, y = FALSE,
       singular.ok = TRUE, contrasts = NULL, offset = NULL,
       control = NULL, init = NULL, ...)
```

Arguments

formula	a symbolic description of the model to be fit. See lm and formula for more details.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>lmrob</code> is called.
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.

na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of <code>options</code> , and is <code>na.fail</code> if that is unset. The “factory-fresh” default is <code>na.omit</code> . Another possible value is <code>NULL</code> , no action. Value <code>na.exclude</code> can be useful.
method	string specifying the estimator-chain. MM is interpreted as SM. See <i>Details</i> .
model, x, y	logicals. If TRUE the corresponding components of the fit (the model frame, the model matrix, the response) are returned.
singular.ok	logical. If FALSE (the default in S but not in R) a singular fit is an error.
contrasts	an optional list. See the <code>contrasts.arg</code> of <code>model.matrix.default</code> .
offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting. An <code>offset</code> term can be included in the formula instead or as well, and if both are specified their sum is used.
control	a <code>list</code> specifying control parameters; use the function <code>lmrob.control(.)</code> and see its help page.
init	an optional argument to specify or supply the initial estimate. See <i>Details</i> .
...	can be used to specify control parameters directly instead of via <code>control</code> .

Details

Overview This function computes an MM-type regression estimator as described in Yohai (1987) and Koller and Stahel (2011). By default it uses a bi-square redescending score function, and it returns a highly robust and highly efficient estimator (with 50% breakdown point and 95% asymptotic efficiency for normal errors). The computation is carried out by a call to `lmrob.fit()`.

The argument setting of `lmrob.control` is provided to set alternative defaults as suggested in Koller and Stahel (2011) (use `setting='KS2011'`). For details, see `lmrob.control`.

Initial Estimator `init`: The initial estimator may be specified using the argument `init`. This argument can either be a string, a function or a list. A *string* can be used to specify built in internal estimators (currently S and M-S, see *See also* below). A *function* taking arguments `x`, `y`, `control`, `mf` (where `mf` stands for `model.frame`) and returning a list containing at least the initial coefficients as `coefficients` and the initial scale estimate `scale`. Or a *list* giving the initial coefficients and scale as `coefficients` and `scale`. See also *Examples*.

Note that if the `init` argument supplied is a function or list, the `method` argument must *not* contain the initial estimator, e.g., use MDM instead of SMDM.

The default, equivalent to `init = "S"`, uses as initial estimator an S-estimator (Rousseeuw and Yohai, 1984) which is computed using the Fast-S algorithm of Salibian-Barrera and Yohai (2006), calling `lmrob.S()`. That function, since March 2012, uses *constrained* subsampling which makes the Fast-S algorithm feasible for categorical data as well.

Method `method`: The following chain of estimates is customizable via the `method` argument of `lmrob.control`. There are currently two types of estimates available,

"M": corresponds to the standard M-regression estimate.

"D": stands for the Design Adaptive Scale estimate as proposed in Koller and Stahel (2011).

The `method` argument takes a string that specifies the estimates to be calculated as a chain. Setting `method='SMDM'` will result in an initial S-estimate, followed by an M-estimate, a Design Adaptive Scale estimate and a final M-step. For methods involving a D-step, the default value of `psi` (see `lmrob.control`) is changed to "lqq".

By default, standard errors are computed using the formulas of Croux, Dhaene and Hoorelbeke (2003) (`lmrob.control` option `cov=".vcov.avar1"`). This method, however, works only for MM-estimates. For other method arguments, the covariance matrix estimate used is based on the asymptotic normality of the estimated coefficients (`cov=".vcov.w"`) as described in Koller and Stahel (2011).

Value

An object of class `lmrob`. A list that includes the following components:

<code>coefficients</code>	The estimate of the coefficient vector
<code>init.S</code>	The list returned by <code>lmrob.S</code> or <code>lmrob.M.S</code> (for MM-estimates only)
<code>init</code>	A similar list that contains the results of intermediate estimates (not for MM-estimates).
<code>scale</code>	The scale as used in the M estimator.
<code>cov</code>	The estimated covariance matrix of the regression coefficients
<code>residuals</code>	Residuals associated with the estimator
<code>fitted.values</code>	Fitted values associated with the estimator
<code>weights</code>	the “robustness weights” $\psi(r_i/S)/(r_i/S)$.
<code>converged</code>	TRUE if the IRWLS iterations have converged

Author(s)

Matias Salibian-Barrera and Manuel Koller

References

- Croux, C., Dhaene, G. and Hoorelbeke, D. (2003) *Robust standard errors for robust estimators*, Discussion Papers Series 03.16, K.U. Leuven, CES.
- Koller, M. and Stahel, W.A. (2011), Sharpening Wald-type inference in robust regression for small samples, *Computational Statistics & Data Analysis* **55**(8), 2504–2515.
- Maronna, R. A., and Yohai, V. J. (2000). Robust regression with both continuous and categorical predictors. *Journal of Statistical Planning and Inference* **89**, 197–214.
- Rousseeuw, P.J. and Yohai, V.J. (1984) Robust regression by means of S-estimators, In *Robust and Nonlinear Time Series*, J. Franke, W. Härdle and R. D. Martin (eds.). Lectures Notes in Statistics 26, 256–272, Springer Verlag, New York.
- Salibian-Barrera, M. and Yohai, V.J. (2006) A fast algorithm for S-regression estimates, *Journal of Computational and Graphical Statistics*, **15**(2), 414–427.
- Yohai, V.J. (1987) High breakdown-point and high efficiency estimates for regression. *The Annals of Statistics* **15**, 642–65.

See Also

`lmrob.control`; for the algorithms `lmrob.S`, `lmrob.M.S` and `lmrob.fit`; and for methods, `predict.lmrob`, `summary.lmrob`, `print.lmrob`, and `plot.lmrob`.

Examples

```

data(coleman)
set.seed(0)
summary( m1 <- lmrob(Y ~ ., data=coleman) )
summary( m2 <- lmrob(Y ~ ., data=coleman, setting = 'KS2011') )

data(starsCYG, package = "robustbase")
## Plot simple data and fitted lines
plot(starsCYG)
  lmST <- lm(log.light ~ log.Te, data = starsCYG)
  (RlmST <- lmrob(log.light ~ log.Te, data = starsCYG))
  abline(lmST, col = "red")
  abline(RlmST, col = "blue")
  summary(RlmST)
  vcov(RlmST)
  stopifnot(all.equal(fitted(RlmST),
                      predict(RlmST, newdata = starsCYG),
                      tol = 1e-14))

## --- init argument
## string
set.seed(0)
m3 <- lmrob(Y ~ ., data=coleman, init = "S")
stopifnot(all.equal(m1[-17], m3[-17]))
## function
initFun <- function(x, y, control, mf) {
  init.S <- lmrob.S(x, y, control)
  list(coefficients=init.S$coef, scale = init.S$scale)
}
set.seed(0)
m4 <- lmrob(Y ~ ., data=coleman, method = "M", init = initFun)
## list
m5 <- lmrob(Y ~ ., data=coleman, method = "M",
            init = list(coefficients = m3$init$coef, scale = m3$scale))
stopifnot(all.equal(m4[-17], m5[-17]))

```

lmrob..D..fit

Compute Design Adaptive Scale estimate

Description

This function calculates a Design Adaptive Scale estimate for a given MM-estimate. This is supposed to be a part of a chain of estimates like SMD or SMDM.

Usage

```
lmrob..D..fit(obj, x=obj$x, control = obj$control)
```

Arguments

obj	lmrob-object based on which the estimate is to be calculated.
x	The design matrix, if missing the method tries to get it from obj\$x and if this fails from obj\$model.
control	list of control parameters, as returned by lmrob.control .

Details

This function is used by [lmrob.fit](#) and typically not to be used on its own.

Value

The given lmrob-object with the following elements updated:

scale	The Design Adaptive Scale estimate
converged	TRUE if the scale calculation converged, FALSE other.

Author(s)

Manuel Koller

References

Koller, M. and Stahel, W.A. (2011), Sharpening Wald-type inference in robust regression for small samples, *Computational Statistics & Data Analysis* **55**(8), 2504–2515.

See Also

[lmrob.fit](#), [lmrob](#)

Examples

```
data(stackloss)
## Compute manual SMD-estimate:
## 1) MM-estimate
m1 <- lmrob(stack.loss ~ ., data = stackloss)
## 2) Add Design Adaptive Scale estimate
m2 <- lmrob..D..fit(m1)
print(c(m1$scale, m2$scale))

summary(m1)
summary(m2) ## the covariance matrix estimate is also updated
```

lmrob..M..fit	<i>Compute M-estimators of regression</i>
---------------	---

Description

This function performs RWLS iterations to find an M-estimator of regression. When started from an S-estimated `beta.initial`, this results in an MM-estimator.

Usage

```
lmrob..M..fit(x, y, beta.initial, scale, control, obj)
```

Arguments

<code>x</code>	design matrix ($n \times p$) typically including a column of 1s for the intercept.
<code>y</code>	numeric response vector (of length n).
<code>beta.initial</code>	numeric vector (of length p) of initial estimate. Usually the result of an S-regression estimator.
<code>scale</code>	robust residual scale estimate. Usually an S-scale estimator.
<code>control</code>	list of control parameters, as returned by <code>lmrob.control</code> . Currently, only the components <code>c("max.it", "rel.tol", "trace.lev", "psi", "tuning.psi", "method")</code> are accessed.
<code>obj</code>	an optional <code>lmrob</code> -object. If specified, this is used to set values for the other arguments.

Details

This function is used by `lmrob.fit` and typically not to be used on its own.

Value

A list with the following elements:

<code>coef</code>	the M-estimator (or MM-estim.) of regression
<code>control</code>	the control list input used
<code>scale</code>	The residual scale estimate
<code>seed</code>	The random number generator seed
<code>converged</code>	TRUE if the RWLS iterations converged, FALSE otherwise

Author(s)

Matias Salibian-Barrera and Martin Maechler

References

Yohai, 1987

See Also

[lmrob.fit](#), [lmrob](#); [rlm](#) from package **MASS**.

Examples

```

data(stackloss)
X <- model.matrix(stack.loss ~ . , data = stackloss)
y <- stack.loss
## Compute manual MM-estimate:
## 1) initial LTS:
m0 <- ltsReg(X[,-1], y)
## 2) M-estimate started from LTS:
m1 <- lmrob..M..fit(X, y, beta.initial = coef(m0), scale = m0$scale,
                    control = lmrob.control(tuning.psi = 1.6,
                                             psi = 'bisquare'))

cbind(m0$coef, m1$coef)
## the scale is kept fixed:
stopifnot(identical(unname(m0$scale), m1$scale))

## robustness weights: are
r.s <- with(m1, residuals/scale) # scaled residuals
m1.wts <- robustbase::tukeyPsi1(r.s, cc = 1.6) / r.s
summarizeRobWeights(m1.wts)
##--> outliers 1,3,4,13,21
which(m0$lts.wt == 0) # 1,3,4,21 but not 13

## Add M-step to SMD-estimate
m2 <- lmrob(stack.loss ~ ., data = stackloss, method = 'SMD')
m3 <- lmrob..M..fit(obj = m2)

## Simple function that allows custom initial estimates
## (Deprecated use init argument to lmrob() instead.)
lmrob.custom <- function(x, y, beta.initial, scale, terms) {
  ## initialize object
  obj <- list(control = lmrob.control("KS2011"),
              terms = terms) ## terms is needed for summary()

  ## M-step
  obj <- lmrob..M..fit(x, y, beta.initial, scale, obj = obj)
  ## D-step
  obj <- lmrob..D..fit(obj, x)
  ## Add some missing elements
  obj$cov <- TRUE ## enables calculation of cov matrix
  obj$p <- obj$qr$rank
  obj$degree.freedom <- length(y) - obj$p
  ## M-step
  obj <- lmrob..M..fit(x, y, obj=obj)
  obj$control$method <- ".MDM"
  obj
}

m4 <- lmrob.custom(X, y, m2$init$init.S$coef,
                  m2$init$scale, m2$terms)

```

```
stopifnot(all.equal(m4$coef, m3$coef))

## Start from ltsReg:
m5 <- ltsReg(stack.loss ~ ., data = stackloss)
m6 <- lmrob.custom(m5$X, m5$Y, coef(m5), m5$scale, m5$terms)
```

lmrob.control

Tuning parameters for lmrob

Description

Tuning parameters for `lmrob`, the MM-type regression estimator and the associated S-, M- and D-estimators. Using `setting="KS2011"` sets the defaults as suggested by Koller and Stahel (2011).

Usage

```
lmrob.control(setting, seed = NULL, nResample = 500,
              tuning.chi = NULL, bb = 0.5, tuning.psi = NULL,
              max.it = 50, groups = 5, n.group = 400,
              k.fast.s = 1, best.r.s = 2, k.max = 200, k.m_s = 20,
              refine.tol = 1e-7, rel.tol = 1e-7, trace.lev = 0,
              mts = 1000, subsampling = c("constrained", "simple"),
              compute.rd = FALSE, method = 'MM',
              psi = c('bisquare', 'lqq', 'welsh', 'optimal', 'hampel',
                    'ggw'), numpoints = 10, cov = '.vcov.avar1',
              split.type = c("f", "fi", "fii"), ...)
```

Arguments

<code>setting</code>	a string specifying alternative default values. Leave empty for the defaults or use "KS2011" for the defaults suggested by Koller and Stahel (2011). See <i>Details</i> .
<code>seed</code>	an integer vector, the seed to be used for random re-sampling used in obtaining candidates for the initial S-estimator; see <code>.Random.seed</code> . The current value of <code>.Random.seed</code> will be preserved if <code>seed</code> is set; otherwise (by default), <code>.Random.seed</code> will be modified as usual from calls to <code>runif()</code> .
<code>nResample</code>	number of re-sampling candidates to be used to find the initial S-estimator. Currently defaults to 500 which works well in most situations (see references).
<code>tuning.chi</code>	tuning constant vector for the S-estimator. Sensible defaults are set according to <code>psi</code> to yield a 50% breakdown estimator. See <i>Details</i> .
<code>bb</code>	expected value under the normal model of the "chi" (rather $\rho(rho)$) function with tuning constant equal to <code>tuning.chi</code> . This is used to compute the S-estimator.
<code>tuning.psi</code>	tuning constant vector for the redescending M-estimator. Depending on the value of <code>psi</code> this constant is set to yield an estimator with asymptotic efficiency of 95% for normal errors. See <i>Details</i> .

max.it	integer specifying the maximum number of IRWLS iterations.
groups	(for the fast-S algorithm): Number of random subsets to use when the data set is large.
n.group	(for the fast-S algorithm): Size of each of the groups above. Note that this must be at least p .
k.fast.s	(for the fast-S algorithm): Number of local improvement steps (“ I -steps”) for each re-sampling candidate.
k.m_s	(for the M-S algorithm): specifies after how many unsuccessful refinement steps the algorithm stops.
best.r.s	(for the fast-S algorithm): Number of of best candidates to be iterated further (i.e., “ <i>refined</i> ”); is denoted t in Salibian-Barrera & Yohai(2006).
k.max	(for the fast-S algorithm): maximal number of refinement steps for the “fully” iterated best candidates.
refine.tol	(for the fast-S algorithm): relative convergence tolerance for the fully iterated best candidates.
rel.tol	(for the RWLS iterations of the MM algorithm): relative convergence tolerance for the parameter vector.
trace.lev	integer indicating if the progress of the MM-algorithm should be traced (increasingly); default trace.lev = 0 does no tracing.
mts	maximum number of samples to try in subsampling algorithm.
subsampling	type of subsampling to be used, simple for simple subsampling (default prior to version 0.9), constrained for constrained subsampling. See lmrob.S .
compute.rd	logical indicating if robust distances (based on the MCD robust covariance estimator covMcd) are to be computed for the robust diagnostic plots. This may take some time to finish, particularly for large data sets, and can lead to singularity problems when there are factor explanatory variables (with many levels, or levels with “few” observations). Hence, is FALSE by default.
method	string specifying the estimator-chain. MM is interpreted as SM. See <i>Details</i> of lmrob for a description of the possible values.
psi	string specifying the type ψ -function used. See <i>Details</i> of lmrob . Defaults to “bisquare” for S and MM-estimates, otherwise “lqq”.
numpoints	Number of points used in Gauss quadrature.
cov	Function or string with function name to be used to calculate covariance matrix estimate. See <i>Details</i> of lmrob .
split.type	Determines how categorical and continuous variables are split. See splitFrame .
...	Further arguments are added to the control list.

Details

The option setting="KS2011" alters the default arguments. They are changed to method = 'SMDM', psi = 'lqq', max.i

The defaults of all the remaining arguments are not changed.

By default, tuning.chi and tuning.psi are set to yield an MM-estimate with break-down point 0.5 and efficiency of 95% at the normal. They are:

	psi	tuning.chi	tuning.psi
bisquare		1.54764	4.685061
welsh		0.5773502	2.11
ggw		c(-0.5, 1.5, NA, 0.5)	c(-0.5, 1.5, 0.95, NA)
lqq		c(-0.5, 1.5, NA, 0.5)	c(-0.5, 1.5, 0.95, NA)
optimal		0.4047	1.060158
hampel		c(1.5, 3.5, 8)*0.2119163	c(1.5, 3.5, 8)*0.9014

The values for the tuning constant for the ggw psi function are hard coded. The constants vector has four elements: minimal slope, b (controlling the bend at the maximum of the curve), efficiency, break-down point. Use NA for an unspecified value, see examples in the tables.

The constants for the hampel psi function are chosen to have a redescending slope of $-1/3$. Constants for a slope of $-1/2$ would be

	psi	tuning.chi	tuning.psi
hampel		c(2, 4, 8)*0.1981319	c(2, 4, 8)*0.690794

Alternative coefficients for an efficiency of 85% at the normal are given in the table below.

	psi	tuning.psi
bisquare		3.443689
welsh		1.456
ggw		c(-0.5, 1.5, 0.85, NA)
optimal		0.8684
hampel (-1/3)		c(1.5, 3.5, 8)*0.5704545
hampel (-1/2)		c(2, 4, 8)*0.4769578

Author(s)

Matias Salibian-Barrera, Martin Maechler and Manuel Koller

References

Koller, M. and Stahel, W.A. (2011), Sharpening Wald-type inference in robust regression for small samples, *Computational Statistics & Data Analysis* **55**(8), 2504–2515.

See Also

[lmrob](#), also for references and examples.

Examples

```
## Show the default settings:
str(lmrob.control())

## Use Koller & Stahel(2011)'s recommendation but a larger 'max.it':
str(ctrl <- lmrob.control("KS2011", max.it = 1000))
```

<code>lmrob.fit</code>	<i>MM-type estimator for regression</i>
------------------------	---

Description

Compute MM-type estimators of regression: An S-estimator is used as starting value, and an M-estimator with fixed scale and redescending psi-function is used from there. Optionally a D-step (Design Adaptive Scale estimate) as well as a second M-step is calculated.

Usage

```
lmrob.fit(x, y, control, init = NULL)
```

Arguments

<code>x</code>	design matrix ($n \times p$) typically including a column of 1s for the intercept.
<code>y</code>	numeric response vector (of length n).
<code>control</code>	a list of control parameters as returned by <code>lmrob.control</code> , used for both the initial S-estimate and the subsequent M- and D-estimates.
<code>init</code>	optional list of initial estimates. See <i>Details</i> .

Details

This function is the basic fitting function for MM-type estimation, called by `lmrob` and typically not to be used on its own.

If given, `init` must be a list of initial estimates containing at least the initial coefficients and scale as coefficients and scale. Otherwise it calls `lmrob.S(.)` and uses it as initial estimator.

Value

A list with components

<code>fitted.values</code>	$X\beta$, i.e., X <code>%%</code> coefficients.
<code>residuals</code>	the raw residuals, $y - \text{fitted.values}$
<code>weights</code>	robustness weights derived from the final M-estimator residuals (even when not converged).
<code>rank</code>	
<code>degree.freedom</code>	$n - \text{rank}$
<code>coefficients</code>	estimated regression coefficient vector
<code>scale</code>	the robustly estimated error standard deviation
<code>cov</code>	variance-covariance matrix of coefficients, if the RWLS iterations have converged
<code>control</code>	

iter	
converged	logical indicating if the RWLS iterations have converged.
init.S	the whole initial S-estimator result, including its own converged flag, see lmrob.S (only for MM-estimates).
init	A similar list that contains the results of intermediate estimates (not for MM-estimates).

Author(s)

Matias Salibian-Barrera, Martin Maechler and Manuel Koller

See Also

[lmrob](#), [lmrob..M..fit](#), [lmrob..D..fit](#), [lmrob.S](#)

 lmrob.lar

Least Absolute Residuals / L1 Regression

Description

This method implements the routine L1 in Barrodale and Roberts (1974), which is based on the simplex method of linear programming. It is a copy of [lmRob.lar](#) in the **robust** package.

Usage

```
lmrob.lar(x, y, control, mf)
```

Arguments

x	numeric matrix for the predictors
y	numeric vector for the response
control	list as returned by lmrob.control
mf	dummy parameter.

Details

This method is used for computing the M-S estimate and typically not to be used on its own.

A description of the Fortran subroutines used can be found in Marazzi (1993). In the book, the main method is named RILARS.

Value

A list that includes the following components:

coef	The L1-estimate of the coefficient vector
scale	The residual scale estimate (mad)
resid	The residuals
iter	The number of iterations required by the simplex algorithm
status	Return status (0: optimal, but non unique solution, 1: optimal unique solution)
converged	Convergence status (always TRUE), needed for lmrob.fit .

Author(s)

Manuel Koller

References

Marazzi, A. (1993). *Algorithms, routines, and S functions for robust statistics*. Wadsworth & Brooks/Cole, Pacific Grove, CA.

See Also

[rq](#) from package **quantreg**.

Examples

```
data(stackloss)
X <- model.matrix(stack.loss ~ . , data = stackloss)
y <- stack.loss
lmrob.lar(X, y)
```

lmrob.M.S

M-S regression estimators

Description

Computes an M-S-estimator for linear regression using the “M-S” algorithm.

Usage

```
lmrob.M.S(x, y, control, mf, split)
```

Arguments

x	numeric matrix for the predictors
y	numeric vector for the response
control	list as returned by lmrob.control
mf	model.frame as returned by model.frame
split	(optional) list as returned by splitFrame

Details

This function is used by `lmrob` and not intended to be used on its own (because an M-S-estimator has too low efficiency ‘on its own’).

An M-S estimator is a combination of an S-estimator for the continuous variables and an L1-estimator for the categorical variables.

The S-estimator is estimated using a subsampling algorithm. If the model includes interactions between categorical (`factor`) and continuous variables, the subsampling algorithm might fail. In this case, one can choose to assign the interaction to the categorical side of variables rather than to the continuous side. This can be accomplished via the control argument `split.type` or by specifying `split`, see `splitFrame`.

Note that the return status `converged` does not refer to the actual convergence status. The algorithm used does not guarantee convergence and thus true convergence is almost never reached. This is, however, not a problem if the estimate is only used as initial estimate part of an MM or SMDM estimate.

Value

A list with components

<code>coefficients</code>	numeric vector (length p) of M-S-regression coefficient estimates.
<code>scale</code>	the M-S-scale residual estimate
<code>residuals</code>	numeric vector (length n) of the residuals.
<code>weights</code>	numeric vector (length n) of the robustness weights.
<code>control</code>	the same list as the <code>control</code> argument.
<code>converged</code>	Convergence status (always TRUE), needed for <code>lmrob.fit</code> .

Author(s)

Manuel Koller

References

Maronna, R. A., and Yohai, V. J. (2000). Robust regression with both continuous and categorical predictors. *Journal of Statistical Planning and Inference* **89**, 197–214.

See Also

`lmrob`; for a description of the available split types, see `splitFrame`.

`lmRob` in package `robust` uses a version of the M-S algorithm automatically when the formula contains factors. Our version however follows Maronna and Yohai (2000) more closely.

Examples

```

data(education)
education <- within(education, Region <- factor(Region))
flm <- lm(Y ~ Region + X1 + X2 + X3, education)
x <- model.matrix(flm)
y <- education$Y # == model.response(model.frame(flm))
set.seed(17)
f.MS <- lmrob.M.S(x, y, control = lmrob.control(),
                 mf = model.frame(flm))

## The typical use of the "M-S" estimator -- as initial estimate :
fmMS <- lmrob(Y ~ Region + X1 + X2 + X3, education,
              init = "M-S")

```

lmrob.S

S-regression estimators

Description

Computes an S-estimator for linear regression, using the “fast S” algorithm.

Usage

```
lmrob.S(x, y, control, trace.lev = control$trace.lev, mf = NULL)
```

Arguments

x	design matrix
y	response vector
control	list as returned by lmrob.control
trace.lev	integer indicating if the progress of the algorithm should be traced (increasingly); default trace.lev = 0 does no tracing.
mf	dummy parameter.

Details

This function is used by [lmrob.fit](#) and not intended to be used on its own (because an S-estimator has too low efficiency ‘on its own’).

By default, the subsampling algorithm uses a customized LU decomposition which ensures a non singular subsample (if this is at all possible). This makes the Fast-S algorithm also feasible for categorical and mixed continuous-categorical data.

One can revert to the old (unconstrained) subsampling scheme by setting the parameter `subsampling` in `control` to `simple`.

Value

A list with components

coefficients	numeric vector (length p) of S-regression coefficient estimates.
scale	the S-scale residual estimate
fitted.values	numeric vector (length n) of the fitted values.
residuals	numeric vector (length n) of the residuals.
weights	numeric vector (length n) of the robustness weights.
k.iter	(maximal) number of refinement iterations used.
converged	logical indicating if all refinement iterations had converged.
control	the same list as the control argument.

Author(s)

Matias Salibian-Barrera and Manuel Koller (and Martin Maechler for minor details)

See Also

[lmrob](#), also for references.

Examples

```
set.seed(33)
x1 <- sort(rnorm(30)); x2 <- sort(rnorm(30)); x3 <- sort(rnorm(30))
X. <- cbind(x1, x2, x3)
y <- 10 + X. %*% (10*(2:4)) + rnorm(30)/10
y[1] <- 500 # a moderate outlier
X.[2,1] <- 20 # an X outlier
X1 <- cbind(1, X.)

(m.lm <- lm(y ~ X.))
set.seed(12)
m.lmS <- lmrob.S(x=X1, y=y,
                 control = lmrob.control(nRes = 20), trace.lev=1)
m.lmS[c("coefficients", "scale")]
all.equal(m.lmS$coef, 10 * (1:4), tol = 0.005, check.attributes = FALSE)
stopifnot(all.equal(m.lmS$coef, 10 * (1:4),
                   tol = 0.005, check.attributes = FALSE),
          all.equal(m.lmS$scale, 1/10, tol = 0.09))
```

los *Length of Stay Data*

Description

Length of stay for 201 patients that stayed at the University Hospital of Lausanne during the year 2000.

Usage

```
data(los)
```

Format

Vector of integer values giving the length of stay (days):

```
int [1:201] 16 13 17 4 15 24 59 18 33 8 ...
```

Details

These data may be used to estimate and predict the total resource consumption of this group of patients.

Cf. Ruffieux, Paccaud and Marazzi (2000).

Source

The data were kindly provided by A. Marazzi.

Cf. Hubert, M. and Vandervieren, E. (2006), p. 13–15.

References

Ruffieux, C., Paccaud, F. and A. Marazzi (2000) Comparing rules for truncating hospital length of stay; *Casemix Quarterly* **2**, n. 1.

Hubert, M. and Vandervieren, E. (2006) *An Adjusted Boxplot for Skewed Distributions*, Technical Report TR-06-11, KU Leuven, Section of Statistics, Leuven.

<http://wis.kuleuven.be/stat/robust/Papers/TR0611.pdf>

Examples

```
summary(los) # quite skewed, with median(.) = 8
plot(table(los))
boxplot(los, horizontal=TRUE, add=TRUE, col = "red", axes=FALSE)
##-> "outliers" instead of "just skewed"

hist(log(los))
boxplot(log(los), add=TRUE, col=2, border=2, horizontal = TRUE, at = -1)

if(R.version$sarch == "x86_64") { ## FIXME -- bug in C code
```

```
## Hubert and Vandervieren (2006), p. 15, Fig. 11.
adjbox(los, col = "gray", outcol = "red")
boxplot(los, add = TRUE, staplewex = 0.2, outcex = 0.5, outpch = 4)
title("adjbox(los) and boxplot(los) superimposed")
}
```

ltsReg

Least Trimmed Squares Robust (High Breakdown) Regression

Description

Carries out least trimmed squares (LTS) robust (high breakdown point) regression.

Usage

```
ltsReg(x, ...)

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

## Default S3 method:
ltsReg(x, y, intercept = TRUE, alpha = 1/2, nsamp = 500,
       adjust = FALSE, mcd = TRUE, qr.out = FALSE, yname = NULL,
       seed = NULL, trace = FALSE, use.correction=TRUE, control, ...)
```

Arguments

formula	a formula of the form $y \sim x_1 + x_2 + \dots$
data	data frame from which variables specified in <code>formula</code> 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 <code>na.action</code> setting of options , and is <code>na.fail</code> if that is unset. The “factory-fresh” default is <code>na.omit</code> . Another possible value is <code>NULL</code> , no action. Value <code>na.exclude</code> 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 <code>contrasts.arg</code> of model.matrix.default .

offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting. An <code>offset</code> term can be included in the formula instead or as well, and if both are specified their sum is used.
x	a matrix or data frame containing the explanatory variables.
y	the response: a vector of length the number of rows of x..
intercept	if true, a model with constant term will be estimated; otherwise no constant term will be included. Default is <code>intercept = TRUE</code>
alpha	the percentage (roughly) of squared residuals whose sum will be minimized, by default 0.5. In general, alpha must between 0.5 and 1.
nsamp	number of subsets used for initial estimates or "best" or "exact". Default is <code>nsamp = 500</code> . For <code>nsamp="best"</code> exhaustive enumeration is done, as long as the number of trials does not exceed 5000. For "exact", exhaustive enumeration will be attempted however many samples are needed. In this case a warning message will be displayed saying that the computation can take a very long time.
adjust	whether to perform intercept adjustment at each step. Since this can be time consuming, the default is <code>adjust = FALSE</code> .
mcd	whether to compute robust distances using Fast-MCD.
qr.out	whether to return the QR decomposition (see <code>qr</code>); defaults to false.
yname	the name of the dependent variable. Default is <code>yname = NULL</code>
seed	initial seed for random generator, see <code>rrcov.control</code> .
trace	logical (or integer) indicating if intermediate results should be printed; defaults to FALSE; values ≥ 2 also produce print from the internal (Fortran) code.
use.correction	whether to use finite sample correction factors. Default is <code>use.correction=TRUE</code>
control	a list with estimation options - same as these provided in the function specification. 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.
...	arguments passed to or from other methods.

Details

The LTS regression method minimizes the sum of the h smallest squared residuals, where $h > n/2$, i.e. at least half the number of observations must be used. The default value of h (when `alpha=1/2`) is roughly $n/2$, more precisely, $(n+p+1) \% \% 2$ where n is the total number of observations, but by setting `alpha`, the user may choose higher values up to n , where $h = h(\alpha, n, p) = \mathbf{h.alpha.n}(\alpha, n, p)$. The LTS estimate of the error scale is given by the minimum of the objective function multiplied by a consistency factor and a finite sample correction factor – see Pison et al. (2002) for details. The rescaling factors for the raw and final estimates are returned also in the vectors `raw.cnp2` and `cnp2` of length 2 respectively. The finite sample corrections can be suppressed by setting `use.correction=FALSE`. The computations are performed using the Fast LTS algorithm proposed by Rousseeuw and Van Driessen (1999).

As always, the formula interface has an implied intercept term which can be removed either by $y \sim x - 1$ or $y \sim 0 + x$. See `formula` for more details.

Value

The function `ltsReg` returns an object of class "lts". The `summary` method function is used to obtain (and print) a summary table of the results, and `plot()` can be used to plot them, see the the specific help pages.

The generic accessor functions `coefficients`, `fitted.values` and `residuals` extract various useful features of the value returned by `ltsReg`.

An object of class `lts` is a `list` containing at least the following components:

<code>crit</code>	the value of the objective function of the LTS regression method, i.e., the sum of the h smallest squared raw residuals.
<code>coefficients</code>	vector of coefficient estimates (including the intercept by default when <code>intercept=TRUE</code>), obtained after reweighting.
<code>best</code>	the best subset found and used for computing the raw estimates, with <code>length(best) == quan = h.alpha</code> .
<code>fitted.values</code>	vector like <code>y</code> containing the fitted values of the response after reweighting.
<code>residuals</code>	vector like <code>y</code> containing the residuals from the weighted least squares regression.
<code>scale</code>	scale estimate of the reweighted residuals.
<code>alpha</code>	same as the input parameter <code>alpha</code> .
<code>quan</code>	the number h of observations which have determined the least trimmed squares estimator.
<code>intercept</code>	same as the input parameter <code>intercept</code> .
<code>cnp2</code>	a vector of length two containing the consistency correction factor and the finite sample correction factor of the final estimate of the error scale.
<code>raw.coefficients</code>	vector of raw coefficient estimates (including the intercept, when <code>intercept=TRUE</code>).
<code>raw.scale</code>	scale estimate of the raw residuals.
<code>raw.resid</code>	vector like <code>y</code> containing the raw residuals from the regression.
<code>raw.cnp2</code>	a vector of length two containing the consistency correction factor and the finite sample correction factor of the raw estimate of the error scale.
<code>lts.wt</code>	vector like <code>y</code> containing weights that can be used in a weighted least squares. These weights are 1 for points with reasonably small residuals, and 0 for points with large residuals.
<code>raw.weights</code>	vector containing the raw weights based on the raw residuals and raw scale.
<code>method</code>	character string naming the method (Least Trimmed Squares).
<code>X</code>	the input data as a matrix (including intercept column if applicable).
<code>Y</code>	the response variable as a vector.

Note

We strongly recommend using `lmrob()` instead of `ltsReg` (*See also* below)!

Author(s)

Valentin Todorov <valentin.todorov@chello.at>, based on work written for S-plus by Peter Rousseeuw and Katrien van Driessen from University of Antwerp.

References

- Peter J. Rousseeuw (1984), Least Median of Squares Regression. *Journal of the American Statistical Association* **79**, 871–881.
- P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*. Wiley.
- P. J. Rousseeuw and K. van Driessen (1999) A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **41**, 212–223.
- Pison, G., Van Aelst, S., and Willems, G. (2002) Small Sample Corrections for LTS and MCD. *Metrika* **55**, 111-123.

See Also

`lmrob.S()` provides a fast S estimator with similar breakdown point as `ltsReg()` but better efficiency.

For data analysis, rather use `lmrob` which is based on `lmrob.S`.

`covMcd`; `summary.lts` for summaries.

The generic functions `coef`, `residuals`, `fitted`.

Examples

```
data(heart)
## Default method works with 'x'-matrix and y-var:
heart.x <- data.matrix(heart[, 1:2]) # the X-variables
heart.y <- heart[,"c1length"]
ltsReg(heart.x, heart.y)

data(stackloss)
ltsReg(stack.loss ~ ., data = stackloss)
```

mc

Medcouple, a Robust Measure of Skewness

Description

Compute the ‘medcouple’, a *robust* concept and estimator of skewness. The medcouple is defined as a scaled median difference of the left and right half of distribution, and hence *not* based on the third moment as the classical skewness.

Usage

```
mc(x, ...)
## Default S3 method:
mc(x, na.rm = FALSE, doReflect = (length(x) <= 100),
    eps1 = .Machine$double.eps, eps2 = .Machine$double.xmin,
    maxit = 100, trace.lev = 0, full.result = FALSE, ...)
```

Arguments

<code>x</code>	a numeric vector
<code>na.rm</code>	logical indicating how missing values (NAs) should be dealt with.
<code>doReflect</code>	logical indicating if the internal MC should also be computed on the <i>reflected</i> sample $-x$, with final result $(mc.(x) - mc.(-x))/2$. This makes sense since the internal MC, <code>mc.()</code> computes the <code>himedian()</code> which can differ slightly from the median.
<code>eps1, eps2</code>	tolerance in the algorithm; only change with care!
<code>maxit</code>	maximul number of iterations; typically a few should be sufficient.
<code>trace.lev</code>	integer specifying how much diagnostic output the algorithm (in C) should produce. No output by default, most output for <code>trace_lev = 3</code> .
<code>full.result</code>	logical indicating if the full return values (from C) should be return, as a list <code>attr(*, "mcComp")</code> .
<code>...</code>	potentially further arguments passed to other methods.

Value

a number between -1 and 1, which is the medcouple, $MC(x)$. For `r <- mc(x, full.result = TRUE, ...)`, then `attr(r, "mcComp")` is a list with components

<code>medc</code>	the medcouple $mc.(x)$.
<code>medc2</code>	the medcouple $mc.(-x)$ if <code>doReflect=TRUE</code> .
<code>eps, eps2</code>	tolerances used.
<code>iter, iter2</code>	number of iterations used.
<code>converged, converged2</code>	logical specifying "convergence".

Author(s)

Guy Brys; modifications by Tobias Verbeke and bug fixes and extensions by Martin Maechler.

References

- Guy Brys, Mia Hubert and Anja Struyf (2004) A Robust Measure of Skewness; *JCGS* **13** (4), 996–1017.
- Mia Hubert and E. Vandervieren (2006) An adjusted boxplot for skewed distributions.

See Also

[Qn](#) for a robust measure of scale (aka "dispersion"), ...

Examples

```
mc(1:5) # 0 for a symmetric sample

x1 <- c(1, 2, 7, 9, 10)
mc(x1) # = -1/3

data(cushny)
mc(cushny) # 0.125

stopifnot(mc(c(-20, -5, -2:2, 5, 20)) == 0,
          mc(x1, doReflect=FALSE) == -mc(-x1, doReflect=FALSE),
          all.equal(mc(x1, doReflect=FALSE), -1/3, tol = 1e-12))
```

milk

Daudin's Milk Composition Data

Description

Daudin et al.(1988) give 8 readings on the composition of 86 containers of milk. They speak about 85 observations, but this can be explained with the fact that observations 63 and 64 are identical (as noted by Rocke (1996)).

The data set was used for analysing the stability of principal component analysis by the bootstrap method. In the same context, but using high breakdown point robust PCA, these data were analysed by Todorov et al.(1994). Atkinson (1994) used these data for illustration of the forward search algorithm for identifying of multiple outliers.

Usage

```
data(milk)
```

Format

A data frame with 86 observations on the following 8 variables, all but the first measure units in *grams / liter*.

X1 density
X2 fat content
X3 protein content
X4 casein content
X5 cheese dry substance measured in the factory
X6 cheese dry substance measured in the laboratory
X7 milk dry substance
X8 cheese product

Source

Daudin, J.J. Duby, C. and Trecourt, P. (1988) Stability of Principal Component Analysis Studied by the Bootstrap Method; *Statistics* **19**, 241–258.

References

Todorov, V., Neyko, N., Neytchev, P. (1994) Stability of High Breakdown Point Robust PCA, in *Short Communications, COMPSTAT'94*; Physica Verlag, Heidelberg.

Atkinson, A.C. (1994) Fast Very Robust Methods for the Detection of Multiple Outliers. *J. Amer. Statist. Assoc.* **89** 1329–1339.

Rocke, D. M. and Woodruff, D. L. (1996) Identification of Outliers in Multivariate Data; *J. Amer. Statist. Assoc.* **91** (435), 1047–1061.

Examples

```
data(milk)
covMcd(milk)
```

nlrob

Robust Fitting of Nonlinear Regression Models

Description

nlrob fits a nonlinear regression model by robust M-estimators, using iterated reweighted least squares (IWLS).

Usage

```
nlrob(formula, data, start, weights = NULL, na.action = na.fail,
      psi = psi.huber, test.vec = c("resid", "coef", "w"), maxit = 20,
      acc = 1e-06, algorithm = "default", control = nls.control(),
      trace = FALSE, ...)
```

```
## S3 method for class 'nlrob'
fitted(object, ...)
## S3 method for class 'nlrob'
residuals(object, type = , ...)
## S3 method for class 'nlrob'
predict(object, newdata, ...)
```

Arguments

formula a nonlinear [formula](#) including variables and parameters of the model, such as $y \sim f(x, \theta)$ (cf. [nls](#)). (For some checks: if $f(\cdot)$ is linear, then we need parentheses, e.g., $y \sim (a + b * x)$.)
Do not use w as variable or parameter name!

data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>nlrob</code> is called.
start	a named numeric vector of starting parameters estimates.
weights	an optional vector of weights to be used in the fitting process (for intrinsic weights, not the weights <code>w</code> used in the iterative (robust) fit). I.e., $\sum(w * e^2)$ is minimized with <code>e = residuals</code> , $e_i = y_i - f(x_{reg_i}, \theta)$, where $f(x, \theta)$ is the nonlinear function, and <code>w</code> are the robust weights from <code>resid * weights</code> .
na.action	a function which indicates what should happen when the data contain NAs. The default action is for the procedure to fail. If NAs are present, use <code>na.exclude</code> to have residuals with <code>length == nrow(data) == length(w)</code> , where <code>w</code> are the weights used in the iterative robust loop. This is better if the explanatory variables in <code>formula</code> are time series (and so the NA location is important). For this reason, <code>na.omit</code> , which leads to omission of cases with missing values on any required variable, is not suitable here since the residuals length is different from <code>nrow(data) == length(w)</code> .
psi	a function (possibly by name) of the form <code>g(x, 'tuning constant(s)', deriv)</code> that for <code>deriv=0</code> returns $\psi(x)/x$ and for <code>deriv=1</code> returns $\psi'(x)$. Tuning constants will be passed in via one or several arguments, depending on the psi function. (See also rlm).
test.vec	character string specifying the convergence criterion. The relative change is tested for residuals with a value of "resid" (the default), for coefficients with "coef", and for weights with "w".
maxit	maximum number of iterations in the robust loop.
acc	convergence tolerance for the robust fit.
algorithm	character string specifying the algorithm to use for <code>nls</code> , see there. The default algorithm is a Gauss-Newton algorithm.
control	an optional list of control settings for <code>nls</code> . See <code>nls.control</code> for the names of the settable control values and their effect.
trace	logical value indicating if a "trace" of the <code>nls</code> iteration progress should be printed. Default is FALSE. If TRUE, in each robust iteration, the residual sum-of-squares and the parameter values are printed at the conclusion of each <code>nls</code> iteration. When the "plinear" algorithm is used, the conditional estimates of the linear parameters are printed after the nonlinear parameters.
...	optional arguments (tuning constants) to be passed to the psi function (see psi above).
object	an R object of class "nlrob", typically resulting from <code>nlrob(...)</code> .
type	a string specifying the <i>type</i> of residuals desired. Currently, "response" and "working" are supported.
newdata	a data frame (or list) with the same names as the original data, see e.g., predict.nls .

Value

nlrob() returns an object of S3 class "nlrob", also inheriting from class "nls", (see [nls](#)).

It is a list with components

FIXME ???

There are methods (at least) for the generic accessor functions [summary](#), [coefficients](#), `fitted.values` and `residuals`.

`residuals(.)`, by default `type = "response"`, returns the residuals e_i , defined above as $e_i = Y_i - f(x_i, \theta)$. These differ from the standardized or weighted residuals which, e.g., are assumed to be normally distributed, and a version of which is returned in `working.residuals` component.

Note

This function used to be named [rnls](#) and has been in package [sfsmisc](#), but will be deprecated and dropped there, eventually.

Author(s)

Andreas Ruckstuhl (inspired by [rlm\(\)](#) and [nls\(\)](#)), in July 1994 for S-plus.

Christian Sangiorgio did the update to R and corrected some errors, from June 2002 to January 2005, and Andreas contributed slight changes and the first methods in August 2005.

Help page, testing, more cleanup, methods: Martin Maechler.

See Also

[nls](#), [rlm](#).

Examples

```
DNase1 <- DNase[ DNase$Run == 1, ]

## note that selfstarting models don't work yet % <<< FIXME !!!

##--- without conditional linearity ---

## classical
fmNase1 <- nls( density ~ Asym/(1 + exp(( xmid - log(conc) )/scal ) ),
               data = DNase1,
               start = list( Asym = 3, xmid = 0, scal = 1 ),
               trace = TRUE )
summary( fmNase1 )

## robust
RmN1 <- nlrob( density ~ Asym/(1 + exp(( xmid - log(conc) )/scal ) ),
              data = DNase1, trace = TRUE,
              start = list( Asym = 3, xmid = 0, scal = 1 ) )
summary( RmN1 )

##--- using conditional linearity ---
```

```

## classical
fm2DNase1 <- nls( density ~ 1/(1 + exp(( xmid - log(conc) )/scal ) ),
                 data = DNase1,
                 start = c( xmid = 0, scal = 1 ),
                 alg = "plinear", trace = TRUE )
summary( fm2DNase1 )

## robust
if(FALSE) { # currently fails % FIXME error in nls's nlsModel.plinear()
frm2DNase1 <- nlrob(density ~ 1/(1 + exp(( xmid - log(conc) )/scal ) ),
                  data = DNase1, start = c( xmid = 0, scal = 1 ),
                  alg = "plinear", trace = TRUE )
summary( frm2DNase1 )
} # not yet

### -- new examples -- "moderate outlier":
DN2 <- DNase1
DN2[10,"density"] <- 2*DN2[10,"density"]

fm3DN2 <- nls(density ~ Asym/(1 + exp(( xmid - log(conc) )/scal ) ),
             data = DN2, trace = TRUE,
             start = list( Asym = 3, xmid = 0, scal = 1 ))

## robust
Rm3DN2 <- nlrob(density ~ Asym/(1 + exp(( xmid - log(conc) )/scal ) ),
              data = DN2, trace = TRUE,
              start = list( Asym = 3, xmid = 0, scal = 1 ))

Rm3DN2
summary(Rm3DN2)

## utility function sfsmisc::lseq() :
lseq <- function (from, to, length)
  2^seq(log2(from), log2(to), length.out = length)
## predict() {and plot}:
h.x <- lseq(min(DN2$conc), max(DN2$conc), length = 100)
nDat <- data.frame(conc = h.x)

h.p <- predict(fm3DN2, newdata = nDat)# classical
h.rp <- predict(Rm3DN2, newdata = nDat)# robust

plot(density ~ conc, data=DN2, log="x",
     main = format(formula(Rm3DN2)))
lines(h.x, h.p, col="blue")
lines(h.x, h.rp, col="magenta")
legend("topleft", c("classical nls()", "robust nlrob()"),
      lwd = 1, col= c("blue", "magenta"), inset = 0.05)

```

Description

A typical medium sized environmental data set with hourly measurements of *NOx* pollution content in the ambient air.

Usage

```
data(NOxEmissions)
```

Format

A data frame with 8088 observations on the following 4 variables.

`julday` day number, a factor with levels 373 ... 730, typically with 24 hourly measurements.

`LN0x` log of hourly mean of NOx concentration in ambient air [ppb] next to a highly frequented motorway.

`LN0xEm` log of hourly sum of NOx emission of cars on this motorway in arbitrary units.

`sqrtWS` Square root of wind speed [m/s].

Details

The original data set had more observations, but with missing values. Here, all cases with missing values were omitted (`na.omit(.)`), and then only those were retained that belonged to days with at least 20 (fully) observed hourly measurements.

Source

René Locher (at ZHAW, Switzerland).

See Also

another NOx dataset, [ambientNOxCH](#).

Examples

```
data(NOxEmissions)
plot(LN0x ~ LN0xEm, data = NOxEmissions, cex = 0.25, col = "gray30")

## Not run: ## these take too much time --
## p = 340 ==> already Least Squares is not fast
(lmNOx <- lm(LN0x ~ ., data = NOxEmissions))
plot(lmNOx) #-> indication of 1 outlier

M.NOx <- MASS::rlm(LN0x ~ ., data = NOxEmissions)
## M-estimation works
## whereas MM-estimation fails:
try(MM.NOx <- MASS::rlm(LN0x ~ ., data = NOxEmissions, method = "MM"))
## namely because S-estimation fails:
try(lts.NOx <- ltsReg(LN0x ~ ., data = NOxEmissions))
try(lmR.NOx <- lmrob(LN0x ~ ., data = NOxEmissions))

## End(Not run)
```

pension

Pension Funds Data

Description

The total 1981 premium income of pension funds of Dutch firms, for 18 Professional Branches, from de Wit (1982).

Usage

```
data(pension)
```

Format

A data frame with 18 observations on the following 2 variables.

Income Premium Income (in millions of guilders)

Reserves Premium Reserves (in millions of guilders)

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.76, table 13.

Examples

```
data(pension)
plot(pension)

summary(lm.p <- lm(Reserves ~., data=pension))
summary(lmR.p <- lmrob(Reserves ~., data=pension))
summary(lts.p <- ltsReg(Reserves ~., data=pension))
abline(lm.p)
abline(lmR.p, col=2)
abline(lts.p, col=2, lty=2)

## MM: "the" solution is much simpler:
plot(pension, log = "xy")
lm.lp <- lm(log(Reserves) ~ log(Income), data=pension)
lmR.lp <- lmrob(log(Reserves) ~ log(Income), data=pension)
plot(log(Reserves) ~ log(Income), data=pension)
## no difference between LS and robust:
abline(lm.lp)
abline(lmR.lp, col=2)
```

 phosphor

Phosphorus Content Data

Description

This dataset investigates the effect from inorganic and organic Phosphorus in the soil upon the phosphorus content of the corn grown in this soil, from Prescott (1975).

Usage

```
data(phosphor)
```

Format

A data frame with 18 observations on the following 3 variables.

inorg Inorganic soil Phosphorus

organic Organic soil Phosphorus

plant Plant Phosphorus content

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*. Wiley, p.156, table 24.

Examples

```
data(phosphor)
plot(phosphor)
summary(lm.phosphor <- lm(plant ~ ., data = phosphor))
summary(lts.phosphor <- ltsReg(plant ~ ., data = phosphor))

phosphor.x <- data.matrix(phosphor[, 1:2])
cPh <- covMcd(phosphor.x)
plot(cPh, "dd")
```

 pilot

Pilot-Plant Data

Description

Pilot-Plant data from Daniel and Wood (1971). The response variable corresponds to the acid content determined by titration and the explanatory variable is the organic acid content determined by extraction and weighing. This data set was analyzed also by Yale and Forsythe (1976).

Usage

```
data(pilot)
```

Format

A data frame with 20 observations on the following 2 variables.

X Organic acid content - extraction

Y Acid content - titration

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, page 21, table 1.

Examples

```
data(pilot)
summary(lm.pilot <- lm(Y ~., data=pilot))
```

plot.lmrob

Plot Method for "lmrob" Objects

Description

Diagnostic plots for elements of class lmrob

Usage

```
## S3 method for class 'lmrob'
plot(x, which = 1:5,
     caption = c("Standardized residuals vs. Robust Distances",
                 "Normal Q-Q vs. Residuals", "Response vs. Fitted Values",
                 "Residuals vs. Fitted Values" , "Sqrt of abs(Residuals) vs. Fitted Values"),
     panel = points, sub.caption = deparse(x$call), main = "",
     compute.MD = TRUE,
     ask = prod(par("mfcol")) < length(which) && dev.interactive(),
     ..., p=0.025)
```

Arguments

x	an object as created by lmrob
which	integer number between 1 and 5 to specify which plot is desired
caption	Caption for the different plots
panel	Panel

main	Main title
sub.caption	sub titles
compute.MD	logical indicating if the robust Mahalanobis distances should be recomputed, using <code>covMcd()</code> when needed, i.e., if which contains 1.
ask	waits for user input before displaying each plot
...	optional arguments for par
p	threshold for distance-distance plot

Details

if `compute.MD = TRUE` and the robust Mahalanobis distances need to be computed, they are stored (“cached”) with the object `x` when this function has been called from top-level.

References

Robust diagnostic plots as in Rousseuw and van Zomeren (1990)

See Also

[lmrob](#), also for examples, [plot.lm](#).

Examples

```
data(starsCYG)
## Plot simple data and fitted lines
plot(starsCYG)
lmST <- lm(log.light ~ log.Te, data = starsCYG)
RlmST <- lmrob(log.light ~ log.Te, data = starsCYG)
RlmST
abline(lmST, col = "red")
abline(RlmST, col = "blue")

op <- par(mfrow = c(2,2), mgp = c(1.5, 0.6, 0), mar = .1+c(3,3,3,1))
plot(RlmST, which = c(1:2, 4:5))
par(op)
```

plot.lts

Robust LTS Regression Diagnostic Plots

Description

Four plots (selectable by which) are currently provided:

1. a plot of the standardized residuals versus their index,
2. a plot of the standardized residuals versus fitted values,
3. a Normal Q-Q plot of the standardized residuals, and
4. a regression diagnostic plot (standardized residuals versus robust distances of the predictor variables).

Usage

```
## S3 method for class 'lts'
plot(x, which = c("all", "rqq", "rindex", "rfit", "rdiag"),
     classic=FALSE, ask=(which=="all" && dev.interactive()), id.n, ...)
```

Arguments

<code>x</code>	a <code>lts</code> object, typically result of <code>ltsReg</code> .
<code>which</code>	string indicating which plot to show. See the <i>Details</i> section for a description of the options. Defaults to "all"..
<code>classic</code>	whether to plot the classical distances too. Default is FALSE..
<code>ask</code>	logical indicating if the user should be <i>asked</i> before each plot, see <code>par(ask=.)</code> . Defaults to <code>which == "all" && dev.interactive()</code> .
<code>id.n</code>	number of observations to be identified by a label starting with the most extreme. Default is the number of identified outliers (can be different for the different plots - see <i>Details</i>).
<code>...</code>	other parameters to be passed through to plotting functions.

Details

This function produces several plots based on the robust and classical regression estimates. Which of them to select is specified by the attribute `which`. The possible options are:

`rqq`: Normal Q-Q plot of the standardized residuals;
`rindex`: plot of the standardized residuals versus their index;
`rfit`: plot of the standardized residuals versus fitted values;
`rdiag`: regression diagnostic plot.

The normal quantile plot produces a normal Q-Q plot of the standardized residuals. A line is drawn which passes through the first and third quantile. The `id.n` residuals with largest distances from this line are identified by labels (the observation number). The default for `id.n` is the number of regression outliers (`lts.wt==0`).

In the Index plot and in the Fitted values plot the standardized residuals are displayed against the observation number or the fitted value respectively. A horizontal dashed line is drawn at 0 and two solid horizontal lines are located at +2.5 and -2.5. The `id.n` residuals with largest absolute values are identified by labels (the observation number). The default for `id.n` is the number regression outliers (`lts.wt==0`).

The regression diagnostic plot, introduced by Rousseeuw and van Zomeren (1990), displays the standardized residuals versus robust distances. Following Rousseeuw and van Zomeren (1990), the horizontal dashed lines are located at +2.5 and -2.5 and the vertical line is located at the upper 0.975 percent point of the chi-squared distribution with `p` degrees of freedom. The `id.n` residuals with largest absolute values and/or largest robust Mahalanobis distances are identified by labels (the observation number). The default for `id.n` is the number of all outliers: regression outliers (`lts.wt==0`) + leverage (bad and good) points ($RD > 0.975$ percent point of the chi-squared distribution with `p` degrees of freedom).

References

P. J. Rousseeuw and van Zomeren, B. C. (1990). Unmasking Multivariate Outliers and Leverage Points. *Journal of the American Statistical Association* **85**, 633–639.

P. J. Rousseeuw and K. van Driessen (1999) A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **41**, 212–223.

See Also

[covPlot](#)

Examples

```
data(hbk)
lts <- ltsReg(Y ~ ., data = hbk)
lts
plot(lts, which = "rqq")
```

plot.mcd	<i>Robust Distance Plots</i>
----------	------------------------------

Description

Shows the Mahalanobis distances based on robust and classical estimates of the location and the covariance matrix in different plots. The following plots are available:

- index plot of the robust and mahalanobis distances
- distance-distance plot
- Chisquare QQ-plot of the robust and mahalanobis distances
- plot of the tolerance ellipses (robust and classic)
- Scree plot - Eigenvalues comparison plot

Usage

```
## S3 method for class 'mcd'
plot(x,
      which = c("all", "dd", "distance", "qqchi2",
               "tolEllipsePlot", "screeplot"),
      classic = FALSE, ask = (which=="all" && dev.interactive()),
      cutoff, id.n, labels.id = rownames(x$X), cex.id = 0.75,
      label.pos = c(4,2), tol = 1e-7, ...)

covPlot(x,
         which = c("all", "dd", "distance", "qqchi2",
                  "tolEllipsePlot", "screeplot"),
         classic = FALSE, ask = (which == "all" && dev.interactive()),
         m.cov = covMcd(x),
```

```
cutoff = NULL, id.n, labels.id = rownames(x), cex.id = 0.75,
label.pos = c(4,2), tol = 1e-07, ...)
```

Arguments

x	For the plot() method, a mcd object, typically result of covMcd. For covPlot(), the numeric data matrix such as the X component as returned from covMcd.
which	string indicating which plot to show. See the <i>Details</i> section for a description of the options. Defaults to "all"..
classic	whether to plot the classical distances too. Defaults to FALSE..
ask	logical indicating if the user should be <i>asked</i> before each plot, see par(ask=.). Defaults to which == "all" && dev.interactive().
cutoff	the cutoff value for the distances.
id.n	number of observations to be identified by a label. If not supplied, the number of observations with distance larger than cutoff is used.
labels.id	vector of labels, from which the labels for extreme points will be chosen. NULL uses observation numbers.
cex.id	magnification of point labels.
label.pos	positioning of labels, for the left half and right half of the graph respectively (used as text(. , pos=*)).
tol	tolerance to be used for computing the inverse, see solve. Defaults to tol = 1e-7.
m.cov	an object similar to those of class "mcd"; however only its components center and cov will be used. If missing, the MCD will be computed (via covMcd()).
...	other parameters to be passed through to plotting functions.

Details

These functions produce several plots based on the robust and classical location and covariance matrix. Which of them to select is specified by the attribute which. The plot method for "mcd" objects is calling covPlot() directly, whereas covPlot() should also be useful for plotting other (robust) covariance estimates. The possible options are:

distance index plot of the robust distances

dd distance-distance plot

qqchi2 a qq-plot of the robust distances versus the quantiles of the chi-squared distribution

tolEllipsePlot a tolerance ellipse plot, via tolEllipsePlot()

screepplot an eigenvalues comparison plot - screepplot

The Distance-Distance Plot, introduced by Rousseeuw and van Zomeren (1990), displays the robust distances versus the classical Mahalanobis distances. The dashed line is the set of points where the robust distance is equal to the classical distance. The horizontal and vertical lines are drawn at values equal to the cutoff which defaults to square root of the 97.5% quantile of a chi-squared distribution with p degrees of freedom. Points beyond these lines can be considered outliers.

References

P. J. Rousseeuw and van Zomeren, B. C. (1990). Unmasking Multivariate Outliers and Leverage Points. *Journal of the American Statistical Association* **85**, 633–639.

P. J. Rousseeuw and K. van Driessen (1999) A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **41**, 212–223.

See Also

[tolEllipsePlot](#)

Examples

```
data(Animals, package = "MASS")
brain <- Animals[c(1:24, 26:25, 27:28),]
mcd <- covMcd(log(brain))

plot(mcd, which = "distance", classic = TRUE)# 2 plots
plot(mcd, which = "dd")
plot(mcd, which = "tolEllipsePlot", classic = TRUE)
op <- par(mfrow = c(2,3))
plot(mcd) ## -> which = "all" (5 plots)
par(op)

## same plots for another robust Cov estimate:
data(hbk)
hbk.x <- data.matrix(hbk[, 1:3])
cOGK <- covOGK(hbk.x, n.iter = 2, sigmamu = scaleTau2,
               weight.fn = hard.rejection)
covPlot(hbk.x, m.cov = cOGK, classic = TRUE)
```

Description

Possum diversity data: As issued from a study of the diversity of possum (arboreal marsupials) in the Montane ash forest (Australia), this dataset was collected in view of the management of hardwood forest to take conservation and recreation values, as well as wood production, into account.

The study is fully described in the two references. The number of different species of arboreal marsupials (possum) was observed on 151 different 3ha sites with uniform vegetation. For each site the nine variable measures (see below) were recorded. The problem is to model the relationship between diversity and these other variables.

Usage

```
data(possumDiv)
```

Format

Two different representations of the same data are available:

`possumDiv` is a data frame of 151 observations of 9 variables, where the last two are factors, `eucalyptus` with 3 levels and `aspect` with 4 levels.

`possum.mat` is a numeric (integer) matrix of 151 rows (observations) and 14 columns (variables) where the last seven ones are 0-1 dummy variables, three (E.*) are coding for the kind of eucalyptus and the last four are 0-1 coding for the aspect factor.

The variables have the following meaning:

Diversity main variable of interest is the number of different species of arboreal marsupial (possum) observed, with values in 0:5.

Shrubs the number of shrubs.

Stumps the number of cut stumps from past logging operations.

Stags the number of stags (hollow-bearing trees).

Bark bark index (integer) vector reflecting the quantity of decorticating bark.

Habitat an integer score indicating the suitability of nesting and foraging habitat for Leadbeater's possum.

BAcacia a numeric vector giving the basal area of acacia species.

eucalyptus a 3-level **factor** specifying the species of eucalypt with the greatest stand basal area. This has the same information as the following three variables

E.regnans 0-1 indicator for Eucalyptus regnans

E.delegatensis 0-1 indicator for Eucalyptus deleg.

E.nitens 0-1 indicator for Eucalyptus nitens

aspect a 4-level **factor** specifying the aspect of the site. It is the same information as the following four variables.

NW-NE 0-1 indicator

NW-SE 0-1 indicator

SE-SW 0-1 indicator

SW-NW 0-1 indicator

Source

Eva Cantoni (2004) Analysis of Robust Quasi-deviances for Generalized Linear Models. *Journal of Statistical Software* **10**, 04, <http://www.jstatsoft.org/v10/i04>

References

Lindenmayer, D. B., Cunningham, R. B., Tanton, M. T., Nix, H. A. and Smith, A. P. (1991) The conservation of arboreal marsupials in the montane ash forests of the central highlands of victoria, south-east australia: III. The habitat requirements of leadbeater's possum *gymnobelideus leadbeateri* and models of the diversity and abundance of arboreal marsupials. *Biological Conservation* **56**, 295–315.

Lindenmayer, D. B., Cunningham, R. B., Tanton, M. T., Smith, A. P. and Nix, H. A. (1990) The conservation of arboreal marsupials in the montane ash forests of the victoria, south-east australia, I. Factors influencing the occupancy of trees with hollows, *Biological Conservation* **54**, 111–131.

See also the references in [glmrob](#).

Examples

```
data(possumDiv)
head(possum.mat)

str(possumDiv)
## summarize all variables as multilevel factors:
summary(as.data.frame(lapply(possumDiv, function(v)
                             if(is.integer(v)) factor(v) else v)))
```

predict.glmrob

Predict Method for Robust GLM ("glmrob") Fits

Description

Obtains predictions and optionally estimates standard errors of those predictions from a fitted *robust* generalized linear model (GLM) object.

Usage

```
## S3 method for class 'glmrob'
predict(object, newdata = NULL,
        type = c("link", "response", "terms"), se.fit = FALSE,
        dispersion = NULL, terms = NULL, na.action = na.pass, ...)
```

Arguments

object a fitted object of class inheriting from "glmrob".

newdata optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.

type	the type of prediction required. The default is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable. Thus for a default binomial model the default predictions are of log-odds (probabilities on logit scale) and type = "response" gives the predicted probabilities. The "terms" option returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. The value of this argument can be abbreviated.
se.fit	logical switch indicating if standard errors are required.
dispersion	the dispersion of the GLM fit to be assumed in computing the standard errors. If omitted, that returned by <code>summary</code> applied to the object is used.
terms	with type="terms" by default all terms are returned. A character vector specifies which terms are to be returned
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	optional further arguments, currently simply passed to <code>predict.lmrob()</code> .

Value

If `se = FALSE`, a vector or matrix of predictions. If `se = TRUE`, a list with components

fit	Predictions
se.fit	Estimated standard errors
residual.scale	A scalar giving the square root of the dispersion used in computing the standard errors.

Author(s)

Andreas Ruckstuhl

See Also

`glmrob()` to fit these robust GLM models, `residuals.glmrob()` and other methods; `predict.lm()`, the method used for a non-robust fit.

Examples

```
data(carrots)
## simplistic testing & training:
i.tr <- sample(24, 20)
fm1 <- glmrob(cbind(success, total-success) ~ logdose + block,
              family = binomial, data = carrots, subset = i.tr)

fm1
predict(fm1, carrots[-i.tr, ]) # --> numeric vector
predict(fm1, carrots[-i.tr, ],
        type="response", se = TRUE)# -> a list
```

```

data(vaso)
Vfit <- glmrob(Y ~ log(Volume) + log(Rate), family=binomial, data=vaso)
newd <- expand.grid(Volume = (V. <- seq(.5, 4, by = 0.5)),
                  Rate = (R. <- seq(.25,4, by = 0.25)))
p <- predict(Vfit, newd)
filled.contour(V., R., matrix(p, length(V.), length(R.)),
              main = "predict(glmrob(., data=vaso))", xlab="Volume", ylab="Rate")

```

predict.lmrob

Predict method for Robust Linear Model ("lmrob") Fits

Description

Predicted values based on robust linear model object.

Usage

```

## S3 method for class 'lmrob'
predict(object, newdata, se.fit = FALSE,
        scale = NULL, df = NULL,
        interval = c("none", "confidence", "prediction"), level = 0.95,
        type = c("response", "terms"), terms = NULL,
        na.action = na.pass, pred.var = res.var/weights, weights = 1, ...)

```

Arguments

object	object of class inheriting from "lmrob"
newdata	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
se.fit	a switch indicating if standard errors are required.
scale	scale parameter for std.err. calculation
df	degrees of freedom for scale
interval	type of interval calculation.
level	tolerance/confidence level
type	Type of prediction (response or model term).
terms	if type="terms", which terms (default is all terms)
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
pred.var	the variance(s) for future observations to be assumed for prediction intervals. See 'Details'.
weights	variance weights for prediction. This can be a numeric vector or a one-sided model formula. In the latter case, it is interpreted as an expression evaluated in newdata
...	further arguments passed to or from other methods.

Value

predict.lmrob produces a vector of predictions or a matrix of predictions and bounds with column names fit, lwr, and upr if interval is set. If se.fit is TRUE, a list with the following components is returned:

fit	vector or matrix as above
se.fit	standard error of predicted means
residual.scale	residual standard deviations
df	degrees of freedom for residual

Author(s)

Andreas Ruckstuhl

See Also

[lmrob](#) and the (non-robust) traditional [predict.lm](#) method.

print.lmrob	<i>Print Method for Objects of Class "lmrob"</i>
-------------	--

Description

Print method for elements of class "lmrob".

Usage

```
## S3 method for class 'lmrob'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	an R object of class lmrob, typically created by lmrob .
digits	number of digits for printing, see digits in options .
...	potentially more arguments passed to methods.

See Also

[lmrob](#), [summary.lmrob](#), [print](#) and [summary](#).

Examples

```
data(coleman)
( m1 <- lmrob(Y ~ ., data=coleman) ) # -> print.lmrob() method
```

psiFunc

Constructor for Objects "Psi Function" Class

Description

psiFunc(.) is a convenience interface to new("psi_func", .), i.e. for constructing objects of class "psi_func".

Usage

```
psiFunc(rho, psi, wgt, Dpsi, Erho = NULL, Epsi2 = NULL, EDpsi = NULL, ...)
```

Arguments

rho, psi, wgt, Dpsi

each a [function](#) of x and tuning parameters typically.

Erho, Epsi2, EDpsi

see [psi_func-class](#), and note that these may change in the future.

...

potential further arguments for specifying tuning parameter names and defaults.

Author(s)

Martin Maechler

See Also

[psi_func-class](#) for the class description.

Examples

```
## classical {trivial, not interesting}:  
F1 <- function(x) rep.int(1, length(x))  
cPsi <- psiFunc(rho = function(x) x^2 / 2, psi = function(x) x,  
               wgt = F1, Dpsi = F1,  
               Erho = function(x) rep.int(1/2, length(x)),  
               Epsi2 = F1, EDpsi = F1)
```

psi_func-class

*Class of "Psi Functions" for M-Estimation***Description**

The class "psi_func" is used to store ψ (*psi*) functions for M-estimation. In particular, an object of the class contains $\rho(x)$ (*rho*), its derivative $\psi(x)$ (*psi*), the weight function $\psi(x)/x$, and first derivative of ψ , $Dpsi = \psi'(x)$.

Objects from the Class

Objects can be created by calls of the form `new("psi_func", ...)`, but preferably by `psiFunc(...)`.

Slots

rho: the $\rho()$ function, an object of class "functionX". This is used to formulate the objective function; $\rho()$ can be regarded as generalized negative log-likelihood.

psi: $\psi()$ is the derivative of ρ , $\psi(x) = \frac{d}{dx}\rho(x)$; also of class "functionX".

wgt: The weight function $\psi(x)/x$, of class "functionX".

Dpsi: the derivative of ψ , $Dpsi(x) = \psi'(x)$; of class "functionX".

tDefs: *named* numeric vector of **tuning parameter Default** values.

Erho: Object of class "functionXa1" ~~

Epsi2: Object of class "functionXa1" ~~

EDpsi: Object of class "functionXa1" ~~

xtras: Potentially further information.

Methods

No methods defined with class "psi_func" in the signature.

Author(s)

Martin Maechler

See Also

[psiFunc](#).

Examples

```
str(huberPsi, give.attr = FALSE)
```

pulpfiber

Pulp Fiber and Paper Data

Description

Measurements of aspects pulp fibers and the paper produced from them. Four properties of each are measured in sixty-two samples.

Usage

```
data(pulpfiber)
```

Format

A data frame with 62 observations on the following 8 variables.

X1 numeric vector of arithmetic fiber length

X2 numeric vector of long fiber fraction

X3 numeric vector of fine fiber fraction

X4 numeric vector of zero span tensile

Y1 numeric vector of breaking length

Y2 numeric vector of elastic modulus

Y3 numeric vector of stress at failure

Y4 numeric vector of burst strength

Details

Cited from the reference article: *The dataset contains measurements of properties of pulp fibers and the paper made from them. The aim is to investigate relations between pulp fiber properties and the resulting paper properties. The dataset contains $n = 62$ measurements of the following four pulp fiber characteristics: arithmetic fiber length, long fiber fraction, fine fiber fraction, and zero span tensile. The four paper properties that have been measured are breaking length, elastic modulus, stress at failure, and burst strength.*

The goal is to predict the $q = 4$ paper properties from the $p = 4$ fiber characteristics.

Author(s)

port to R and this help page: Martin Maechler

Source

Rousseeuw, P. J., Van Aelst, S., Van Driessen, K., and Agulló, J. (2004) Robust multivariate regression; *Technometrics* **46**, 293–305.

<http://allserv.ugent.be/~svaelst/data/pulpfiber.txt>

References

Lee, J. (1992) *Relationships Between Properties of Pulp-Fibre and Paper*, unpublished doctoral thesis, U. Toronto, Faculty of Forestry.

Examples

```
data(pulpfiber)
str(pulpfiber)

pairs(pulpfiber, gap=.1)
## 2 blocks of 4 ..
c1 <- cov(pulpfiber)
cR <- covMcd(pulpfiber)
## how different are they: The robust estimate has more clear high correlations:
symnum(cov2cor(c1))
symnum(cov2cor(cR$cov))
```

Qn

Robust Location-Free Scale Estimate More Efficient than MAD

Description

Compute the robust scale estimator Q_n , an efficient alternative to the MAD.

See the references for more.

Usage

```
Qn(x, constant = 2.21914, finite.corr = missing(constant))
```

```
s_Qn(x, mu.too = FALSE, ...)
```

Arguments

x	numeric vector of observations.
constant	number by which the result is multiplied; the default achieves consistency for normally distributed data. Note that until Nov. 2010, “thanks” to a typo in the very first papers, a slightly wrong default constant, 2.2219, was used instead of the correct one which is equal to $1 / (\sqrt{2}) * qnorm(5/8)$. If you need the old slightly off version for historical reproducibility, you can use <code>Qn.old()</code> . Note that the relative difference is only about 1 in 1000, and that the correction should not affect the finite sample corrections for $n \leq 9$.
finite.corr	logical indicating if the finite sample bias correction factor should be applied. Defaults to TRUE unless constant is specified.
mu.too	logical indicating if the <code>median(x)</code> should also be returned for <code>s_Qn()</code> .
...	potentially further arguments for <code>s_Qn()</code> passed to <code>Qn()</code> .

Details

As the (default, consistency) constant needed to be corrected, the finite sample correction has been based on a much more extensive simulation, and on a 3rd or 4th degree polynomial model in $1/n$ for odd or even n , respectively.

Value

`Qn()` returns a number, the Q_n robust scale estimator, scaled to be consistent for σ^2 and i.i.d. Gaussian observations, optionally bias corrected for finite samples.

`s_Qn(x, mu.too=TRUE)` returns a length-2 vector with location (μ) and scale; this is typically only useful for `covOGK(*, sigmamu = s_Qn)`.

Author(s)

Original Fortran code: Christophe Croux and Peter Rousseeuw <rousse@wins.uia.ac.be>.
Port to C and R: Martin Maechler, <maechler@R-project.org>

References

Rousseeuw, P.J. and Croux, C. (1993) Alternatives to the Median Absolute Deviation, *Journal of the American Statistical Association* **88**, 1273–1283.

Christophe Croux and Peter J. Rousseeuw (1992) Time-Efficient Algorithms for Two Highly Robust Estimators of Scale, *Computational Statistics, Vol. 1*, ed. Dodge and Whittaker, Physica-Verlag Heidelberg, 411–428;
also available from <http://win-www.uia.ac.be/u/statis/abstract/Timeff92.htm>.

See Also

`mad` for the ‘most robust’ but much less efficient scale estimator; `Sn` for a similar faster but less efficient alternative; `scaleTau2`.

Examples

```
set.seed(153)
x <- sort(c(rnorm(80), rt(20, df = 1)))
s_Qn(x, mu.too = TRUE)
Qn(x, finite.corr = FALSE)
```

radarImage

Satellite Radar Image Data from near Munich

Description

The data were supplied by A. Frery. They are a part of a synthetic aperture satellite radar image corresponding to a suburb of Munich. Provided are coordinates and values corresponding to three frequency bands for each of 1573 pixels.

Usage

```
data(radarImage)
```

Format

A data frame with 1573 observations on the following 5 variables.

X.coord a numeric vector

Y.coord a numeric vector

Band.1 a numeric vector

Band.2 a numeric vector

Band.3 a numeric vector

Source

The website accompanying the MMY-book: http://www.wiley.com/legacy/wileychi/robust_statistics

Examples

```
data(radarImage)
plot(Y.coord ~ X.coord, data = radarImage)
## see outliers
pairs(radarImage[, 3:5], main = "radarImage (n = 1573)")
```

residuals.glmrob

Residuals of Robust Generalized Linear Model Fits

Description

Compute residuals of a fitted `glmrob` model, i.e., robust generalized linear model fit.

Usage

```
## S3 method for class 'glmrob'
residuals(object,
           type = c("deviance", "pearson", "working",
                   "response", "partial"),
           ...)
```

Arguments

`object` an object of class `glmrob`, typically the result of a call to `glmrob`.

`type` the type of residuals which should be returned. The alternatives are: "deviance" (default), "pearson", "working", "response", and "partial".

... further arguments passed to or from other methods.

Details

The references in [glm](#) define the types of residuals: Davison & Snell is a good reference for the usages of each.

The partial residuals are a matrix of working residuals, with each column formed by omitting a term from the model.

The residuals (S3) method (see [methods](#)) for [glmrob](#) models has been modeled to follow closely the method for classical (non-robust) [glm](#) fitted models. Possibly, see its documentation, i.e., [residuals.glm](#), for further details.

References

See those for the classical GLM's, [glm](#).

See Also

[glmrob](#) for computing object, [anova.glmrob](#); the corresponding *generic* functions, [summary.glmrob](#), [coef](#), [fitted](#), [residuals](#).

Examples

```
### ----- Gamma family -- data from example(glm) ---
clotting <- data.frame(
  u = c(5,10,15,20,30,40,60,80,100),
  lot1 = c(118,58,42,35,27,25,21,19,18),
  lot2 = c(69,35,26,21,18,16,13,12,12))
summary(cl <- glm (lot1 ~ log(u), data=clotting, family=Gamma))
summary(ro <- glmrob(lot1 ~ log(u), data=clotting, family=Gamma))
clotM5.high <- within(clotting, { lot1[5] <- 60 })
cl5.high <- glm (lot1 ~ log(u), data=clotM5.high, family=Gamma)
ro5.high <- glmrob(lot1 ~ log(u), data=clotM5.high, family=Gamma)
rr <- range(residuals(ro), residuals(cl), residuals(ro5.high))
plot(residuals(ro5.high) ~ residuals(cl5.high), xlim = rr, ylim = rr, asp = 1)
abline(0,1, col=2, lty=3)
points(residuals(ro) ~ residuals(cl), col = "gray", pch=3)

## Show all kinds of residuals:
r.types <- c("deviance", "pearson", "working", "response")
sapply(r.types, residuals, object = ro5.high)
```

```
rrcov.control
```

```
Control object for the estimation parameters
```

Description

Auxiliary function for passing the estimation options as parameters to the estimation functions.

NOTE: The name WILL change !!!!

Usage

```
rrcov.control(alpha = 1/2, nsamp = 500, nmini = 300,
              seed = NULL, tolSolve = 1e-14, trace = FALSE,
              use.correction = TRUE, adjust = FALSE)
```

Arguments

alpha	This parameter controls the size of the subsets over which the determinant is minimized, i.e., $\alpha \times n$ observations are used for computing the determinant. Allowed values are between 0.5 and 1 and the default is 0.5.
nsamp	number of subsets used for initial estimates or "best" or "exact". Default is <code>nsamp = 500</code> . If <code>nsamp="best"</code> exhaustive enumeration is done, as far as the number of trials do not exceed 5000. If <code>nsamp="exact"</code> exhaustive enumeration will be attempted however many samples are needed. In this case a warning message will be displayed saying that the computation can take a very long time.
nmini	for <code>covMcd</code> : For large n , the algorithm splits the data into maximally $k_{\text{rep}} = 5$ subsets of size <code>nmini</code> .
seed	initial seed for R's random number generator; see <code>.Random.seed</code> and the description of the <code>seed</code> argument in <code>lmrob.control</code> .
tolSolve	numeric tolerance to be used for inversion (<code>solve</code>) of the covariance matrix in <code>mahalanobis</code> .
trace	whether to print intermediate results. Default is <code>trace = FALSE</code>
use.correction	whether to use finite sample correction factors. Defaults to <code>TRUE</code> .
adjust	whether to perform intercept adjustment at each step. Because this can be quite time consuming, the default is <code>adjust = FALSE</code> .

Details

For details about the estimation options see the corresponding estimation functions.

Value

A list with components, as the parameters passed by the invocation

Author(s)

Valentin Todorov

Examples

```
data(Animals, package = "MASS")
brain <- Animals[c(1:24, 26:25, 27:28),]
data(hbk)
hbkm <- data.matrix(hbk[, 1:3])

ctrl <- rrcov.control(alpha=0.75, trace=TRUE)
```

```
covMcd(hbk.x,      control = ctrl)
covMcd(log(brain), control = ctrl)
```

salinity

Salinity Data

Description

This is a data set consisting of measurements of water salinity (i.e., its salt concentration) and river discharge taken in North Carolina's Pamlico Sound; This dataset was listed by Ruppert and Carroll (1980). In Carrol and Ruppert (1985) the physical background of the data is described. They indicated that observations 5 and 16 correspond to periods of very heavy discharge and showed that the discrepant observation 5 was masked by observations 3 and 16, i.e., only after deletion of these observations it was possible to identify the influential observation 5.

This data set is a prime example of the masking effect.

Usage

```
data(salinity)
```

Format

A data frame with 28 observations on the following 4 variables.

X1 Lagged Salinity

X2 Trend

X3 Discharge

Y Salinity

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.82, table 5.

Examples

```
data(salinity)
summary(lm.sali <- lm(Y ~ . , data = salinity))
summary(rlm.sali <- MASS::rlm(Y ~ . , data = salinity))
summary(lts.sali <- ltsReg(Y ~ . , data = salinity))

salinity.x <- data.matrix(salinity[, 1:3])
c_sal <- covMcd(salinity.x)
plot(c_sal, "tolEllipsePlot")
```

 scaleTau2

Robust Tau-Estimate of Scale

Description

Computes the robust τ -estimate of univariate scale, as proposed by Maronna and Zamar (2002); improved by a consistency factor.

Usage

```
scaleTau2(x, c1 = 4.5, c2 = 3.0, consistency = TRUE,
          mu.too = FALSE, ...)
```

Arguments

x	numeric vector
c1, c2	non-negative numbers, specifying cutoff values for the biweighting of the mean and the rho function respectively.
mu.too	logical indicating if both location and scale should be returned or just the scale (when mu.too=FALSE as by default).
consistency	logical indicating if the consistency correction factor (for the scale) should be applied.
...	potentially additional arguments which are not used.

Details

First, $s_0 := \text{MAD}$, i.e. the equivalent of `mad(x, constant=1)` is computed. Robustness weights $w_i := w_{c1}((x_i - \text{med}(X))/s_0)$ are computed, where $w_c(u) = \max(0, (1 - (u/c)^2)^2)$. The robust location measure is defined as $\mu(X) := (\sum_i w_i x_i) / (\sum_i w_i)$, and the robust τ -estimate is $s(X)^2 := s_0^2 * (1/n) \sum_i \rho_{c2}((x_i - \mu(X))/s_0)$, where $\rho_c(u) = \min(c^2, u^2)$. `scaleTau2(*, consistency=FALSE)` returns $s(X)$, whereas this value is divided by its asymptotic limit when `consistency = TRUE` as by default.

Note that for `n = length(x) == 2`, all equivariant scale estimates are proportional, and specifically, `scaleTau2(x, consistency=FALSE) == mad(x, constant=1)`. See also the reference.

Value

numeric vector of length one (if `mu.too` is FALSE as by default) or two (when `mu.too = TRUE`) with robust scale or (location,scale) estimators $\hat{\sigma}(x)$ or $(\hat{\mu}(x), \hat{\sigma}(x))$.

Author(s)

Original by Kjell Konis with substantial modifications by Martin Maechler.

References

Maronna, R.A. and Zamar, R.H. (2002) Robust estimates of location and dispersion of high-dimensional datasets; *Technometrics* **44**(4), 307–317.

See Also

[Sn](#), [Qn](#), [mad](#); further [covOGK](#) for which `scaleTau2` was designed.

Examples

```
x <- c(1:7, 1000)
sd(x) # non-robust std.deviation
scaleTau2(x)
scaleTau2(x, mu.too = TRUE)
```

SiegelsEx

Siegel's Exact Fit Example Data

Description

A small counterexample data set devised by Andrew Siegel. Six (out of nine) data points lie on the line $y = 0$ such that some robust regression estimators exhibit the “*exact fit*” property.

Usage

```
data(SiegelsEx)
```

Format

A data frame with 9 observations on the following 2 variables.

x a numeric vector

y a numeric vector

Source

Emerson and Hoaglin (1983, p.139)

References

Peter J. Rousseeuw and Annick M. Leroy (1987) *Robust Regression and Outlier Detection* Wiley, p.60–61

Examples

```
data(SiegelsEx)
plot(SiegelsEx, main = "Siegel's example for 'exact fit'")
abline(lm(y ~ x, data = SiegelsEx))
abline(MASS::lqs(y ~ x, data = SiegelsEx, method = "lms"), col = 2)
legend("topright", leg = c("lm", "LMS"), col=1:2, lwd=1, inset = 1/20)
```

Sn *Robust Location-Free Scale Estimate More Efficient than MAD*

Description

Compute the robust scale estimator S_n , an efficient alternative to the MAD.

Usage

```
Sn(x, constant = 1.1926, finite.corr = missing(constant))
```

```
s_Sn(x, mu.too = FALSE, ...)
```

Arguments

x	numeric vector of observations.
constant	number by which the result is multiplied; the default achieves consistency for normally distributed data.
finite.corr	logical indicating if the finite sample bias correction factor should be applied. Default to TRUE unless constant is specified.
mu.too	logical indicating if the <code>median(x)</code> should also be returned for <code>s_Sn()</code> .
...	potentially further arguments for <code>s_Sn()</code> passed to <code>Sn()</code> .

Details

..... FIXME

Value

`Sn()` returns a number, the S_n robust scale estimator, scaled to be consistent for σ^2 and i.i.d. Gaussian observations, optionally bias corrected for finite samples.

`s_Sn(x, mu.too=TRUE)` returns a length-2 vector with location (μ) and scale; this is typically only useful for `covOGK(*, sigmamu = s_Sn)`.

Author(s)

Original Fortran code: Christophe Croux and Peter Rousseeuw <rousse@wins.uia.ac.be>.
Port to C and R: Martin Maechler, <maechler@R-project.org>

References

Rousseeuw, P.J. and Croux, C. (1993) Alternatives to the Median Absolute Deviation, *Journal of the American Statistical Association* **88**, 1273–1283.

See Also

[mad](#) for the ‘most robust’ but much less efficient scale estimator; [Qn](#) for a similar more efficient but slower alternative; [scaleTau2](#).

Examples

```
x <- c(1:10, 100+1:9)# 9 outliers out of 19
Sn(x)
Sn(x, c=1)# 9
Sn(x[1:18], c=1)# 9
set.seed(153)
x <- sort(c(rnorm(80), rt(20, df = 1)))
s_Sn(x, mu.too=TRUE)
```

splitFrame

Split Continuous and Categorical Predictors

Description

Splits the design matrix into categorical and continuous predictors. Categorical variables are variables that are factors or ordered factors.

Usage

```
splitFrame(mf, x = model.matrix(mt, mf),
           type = c("f", "fi", "fii"))
```

Arguments

mf	model frame (as returned by model.frame).
x	(optional) design matrix, defaulting to the derived model.matrix .
type	a character string specifying the split type (see details).

Details

Which split type is used can be controlled with the setting `split.type` in [lmrob.control](#).

There are three split types. The only differences between the types are how interactions between categorical and continuous variables are handled. The extra types of splitting can be used to avoid *Too many singular resamples* errors.

Type "f", the default, assigns only the intercept, categorical and interactions of categorical variables to x1. Interactions of categorical and continuous variables are assigned to x2.

Type "fi" assigns also interactions between categorical and continuous variables to x1.

Type "fii" assigns not only interactions between categorical and continuous variables to x1, but also the (corresponding) continuous variables themselves.

Value

A list that includes the following components:

x1	design matrix containing only categorical variables
x1.idx	logical vectors of the variables considered categorical in the original design matrix
x2	design matrix containing the continuous variables

Author(s)

Manuel Koller

References

Maronna, R. A., and Yohai, V. J. (2000). Robust regression with both continuous and categorical predictors. *Journal of Statistical Planning and Inference* **89**, 197–214.

See Also

[lmrob.M.S](#)

Examples

```
data(education)
education <- within(education, Region <- factor(Region))

## no interactions -- same split for all types:
fm1 <- lm(Y ~ Region + X1 + X2 + X3, education)
spl1 <- splitFrame(fm1$model)
str(spl1)

## with interactions:
fm2 <- lm(Y ~ Region:X1:X2 + X1*X2, education)
s1 <- splitFrame(fm2$model, type="f" )
s2 <- splitFrame(fm2$model, type="fi" )
s3 <- splitFrame(fm2$model, type="fii")
cbind(s1$x1.idx,
      s2$x1.idx,
      s3$x1.idx)
rbind(p.x1 = c(ncol(s1$x1), ncol(s2$x1), ncol(s3$x1)),
      p.x2 = c(ncol(s1$x2), ncol(s2$x2), ncol(s3$x2)))
```

`starsCYG`*Hertzsprung-Russell Diagram Data of Star Cluster CYG OB1*

Description

Data for the Hertzsprung-Russell Diagram of the Star Cluster CYG OB1, which contains 47 stars in the direction of Cygnus, from C.Doom. The first variable is the logarithm of the effective temperature at the surface of the star (T_e) and the second one is the logarithm of its light intensity (L/L_0).

In the Hertzsprung-Russell diagram, which is the scatterplot of these data points, where the log temperature is plotted from left to right, two groups of points are seen: the majority which tend to follow a steep band and four stars in the upper corner. In the astronomy the 43 stars are said to lie on the main sequence and the four remaining stars are called “giants” (the points 11, 20, 30, 34).

Usage

```
data(starsCYG)
```

Format

A data frame with 47 observations on the following 2 variables

`log.Te` Logarithm of the effective temperature at the surface of the star (T_e).

`log.light` Logarithm of its light intensity (L/L_0)

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, p.27, table 3.

Examples

```
data(starsCYG)
plot(starsCYG)
cst <- covMcd(starsCYG)
lm.stars <- lm(log.light ~ log.Te, data = starsCYG)
summary(lm.stars)
plot(lm.stars)
lts.stars <- ltsReg(log.light ~ log.Te, data = starsCYG)
plot(lts.stars)
```

summarizeRobWeights *Print a Nice "summary" of Robustness Weights*

Description

Print a nice “summary” about a numeric vector of robustness weights. Observations with weights around zero are marked as outliers.

Usage

```
summarizeRobWeights(w, digits = getOption("digits"),
                    header = "Robustness weights:",
                    eps = 0.1 / length(w), eps1 = 1e-3, ...)
```

Arguments

w	numeric vector of robustness weights.
digits	digits to be used for printing .
header	string to be printed as header line.
eps	numeric tolerance ϵ : values of w with $ w_i < \epsilon/n$ are said to be outliers.
eps1	numeric tolerance: values of w with $ 1 - w_i < eps1$ are said to have weight ‘ ≈ 1 ’.
...	potential further arguments, passed to print() .

Value

none; the function is used for its side effect of printing.

Author(s)

Martin Maechler

See Also

The [summary](#) methods for [lmrob](#) and [glmrob](#) make use of `summarizeRobWeights()`.

Examples

```
w <- c(1,1,1,1,0,1,1,1,1,0,1,1,.9999,.99999, .5,.6,1e-12)
summarizeRobWeights(w) # two outside  $\approx \{0,1\}$ 
summarizeRobWeights(w, eps1 = 5e-5)# now three outside  $\{0,1\}$ 

## See the summary(<lmrob>) outputs
```

Description

The summary method for class `"glmrob"` summarizes robust fits of (currently only discrete) generalized linear models.

Usage

```
## S3 method for class 'glmrob'
summary(object, correlation = FALSE, symbolic.cor = FALSE, ...)
## S3 method for class 'glmrob'
vcov(object, ...)

## S3 method for class 'summary.glmrob'
print(x, digits = max(3, getOption("digits") - 3),
      symbolic.cor = x$symbolic.cor,
      signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

<code>object</code>	an object of class <code>"glmrob"</code> , usually, a result of a call to <code>glmrob</code> .
<code>correlation</code>	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
<code>symbolic.cor</code>	logical. If TRUE, print the correlations in a symbolic form (see <code>symnum</code>) rather than as numbers.
<code>...</code>	further arguments passed to or from other methods.
<code>x</code>	an object of class <code>"summary.glmrob"</code> .
<code>digits</code>	the number of digits to use for printing.
<code>signif.stars</code>	logical indicating if the P-values should be visualized by so called "significance stars".

Details

`summary.glmrob` returns an object of class `"summary.glmrob"`.

Its `print()` method tries to be smart about formatting the coefficients, standard errors, etc, and gives "significance stars" if `signif.stars` is TRUE (as per default when `options` where not changed).

Value

The function `summary.glmrob` computes and returns a list of summary statistics of the robustly fitted linear model given in `object`. The following elements are in the list:

... FIXME

Author(s)

Andreas Ruckstuhl

See Also[glmrob](#); the generic [summary](#) and also [summary.glm](#).**Examples**

```
data(epilepsy)
Rmod <- glmrob(Ysum ~ Age10 + Base4*Trt, family = poisson,
              data = epilepsy, method= "Mqle")
ss <- summary(Rmod)
ss ## calls print.summary.glmrob()
str(ss) ## internal STRucture of summary object
```

summary.lmrob

*Summary Method for "lmrob" Objects***Description**

Summary method for R object of class "lmrob" and [print](#) method for the summary object.

Further, methods [fitted\(\)](#), [residuals\(\)](#) or [weights\(\)](#) work (via the default methods), and [predict\(\)](#) (see [predict.lmrob](#), [vcov\(\)](#), [model.matrix\(\)](#)) have explicitly defined lmrob methods.

Usage

```
## S3 method for class 'lmrob'
summary(object, correlation = FALSE,
        symbolic.cor = FALSE, ...)
## S3 method for class 'summary.lmrob'
print(x, digits = max(3, getOption("digits") - 3),
      symbolic.cor= x$symbolic.cor,
      signif.stars = getOption("show.signif.stars"), ...)

## S3 method for class 'lmrob'
vcov(object, cov = object$control$cov, ...)
## S3 method for class 'lmrob'
model.matrix(object, ...)
```

Arguments

object an R object of class `lmrob`, typically created by [lmrob](#).

correlation logical variable indicating whether to compute the correlation matrix of the estimated coefficients.

symbolic.cor	logical indicating whether to use symbols to display the above correlation matrix.
x	an R object of class <code>summary.lmrob</code> , typically resulting from <code>summary(lmrob(. .), . .)</code> .
digits	number of digits for printing, see <code>digits</code> in options .
signif.stars	logical variable indicating whether to use stars to display different levels of significance in the individual t-tests.
cov	covariance estimation function to use.
...	potentially more arguments passed to methods.

See Also

[lmrob](#), [predict.lmrob](#), [summary.lm](#), [print](#), [summary](#).

Examples

```
mod1 <- lmrob(stack.loss ~ ., data = stackloss)
sa <- summary(mod1) # calls summary.lmrob(...)
sa                # dispatches to call print.summary.lmrob(...)

## correlation between estimated coefficients:
cov2cor(vcov(mod1))

cbind(fit = fitted(mod1), resid = residuals(mod1),
      wgt= weights(mod1),
      predict(mod1, interval="prediction"))

data(heart)
sm2 <- summary( m2 <- lmrob(clength ~ ., data = heart) )
sm2
```

summary.lts

Summary Method for LTS objects

Description

summary method for class "lts".

Usage

```
## S3 method for class 'lts'
summary(object, correlation = FALSE, ...)

## S3 method for class 'summary.lts'
print(x, digits = max(3, getOption("digits") - 3),
      signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	an object of class "lts", usually, a result of a call to <code>ltsReg</code> .
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
x	an object of class "summary.lts", usually, a result of a call to <code>summary.lts</code> .
digits	the number of significant digits to use when printing.
signif.stars	logical indicating if "significance stars" should be printed, see <code>printCoefmat</code> .
...	further arguments passed to or from other methods.

Details

These functions compute and print summary statistics for weighted least square estimates with weights based on LTS estimates. Therefore the statistics are similar to those for LS but all terms are multiplied by the corresponding weight.

Correlations are printed to two decimal places: to see the actual correlations print `summary(object)$correlation` directly.

Value

The function `summary.lts` computes and returns a list of summary statistics of the fitted linear model given in `object`, using the components of this object (list elements).

residuals	the residuals - a vector like the response <code>y</code> containing the residuals from the weighted least squares regression.
coefficients	a $p \times 4$ matrix with columns for the estimated coefficient, its standard error, t-statistic and corresponding (two-sided) p-value.
sigma	the estimated scale of the reweighted residuals

$$\hat{\sigma}^2 = \frac{1}{n-p} \sum_i R_i^2,$$

where R_i is the i -th residual, `residuals[i]`.

df	degrees of freedom, a 3-vector $(p, n-p, p^*)$, the last being the number of non-aliased coefficients.
fstatistic	(for models including non-intercept terms) a 3-vector with the value of the F-statistic with its numerator and denominator degrees of freedom.
r.squared	R^2 , the "fraction of variance explained by the model",

$$R^2 = 1 - \frac{\sum_i R_i^2}{\sum_i (y_i - y^*)^2},$$

where y^* is the mean of y_i if there is an intercept and zero otherwise.

adj.r.squared	the above R^2 statistic "adjusted", penalizing for higher p .
cov.unscaled	a $p \times p$ matrix of (unscaled) covariances of the $\hat{\beta}_j, j = 1, \dots, p$.
correlation	the correlation matrix corresponding to the above <code>cov.unscaled</code> , if <code>correlation = TRUE</code> is specified.

See Also

[ltsReg](#); the generic [summary](#).

Examples

```
data(Animals2)
ltsA <- ltsReg(log(brain) ~ log(body), data = Animals2)
(slts <- summary(ltsA))
## non-default options for printing the summary:
print(slts, digits = 5, signif.stars = FALSE)
```

summary.mcd

Summary Method for MCD objects

Description

[summary](#) method for class "mcd".

Usage

```
## S3 method for class 'mcd'
summary(object, ...)
## S3 method for class 'summary.mcd'
print(x, digits = max(3, getOption("digits") - 3),
      print.gap = 2, ...)
```

Arguments

object,x	an object of class "mcd" (or "summary.mcd"); usually, a result of a call to covMcd .
digits	the number of significant digits to use when printing.
print.gap	number of horizontal spaces between numbers; see also print.default .
...	further arguments passed to or from other methods.

Details

`summary.mcd()`, the S3 method, simply returns an (S3) object of class "summary.mcd" for which there's a [print](#) method:

`print.summary.mcd` prints summary statistics for the weighted covariance matrix and location estimates with weights based on MCD estimates. While the function `print.mcd` prints only the robust estimates of the location and the covariance matrix, `print.summary.mcd` will print also the correlation matrix (if requested in the call to `covMcd` with `cor=TRUE`), the eigenvalues of the covariance or the correlation matrix and the robust ("Mahalanobis") distances.

Value

summary.mcd returns an summary.mcd object, whereas the print methods returns its first argument via `invisible`, as all print methods do.

See Also

[covMcd](#), [summary](#)

Examples

```
data(Animals, package = "MASS")
brain <- Animals[c(1:24, 26:25, 27:28),]
lbrain <- log(brain)
summary(cLB <- covMcd(lbrain))
```

summary.nlrob

Summarizing Robust Fits of Nonlinear Regression Models

Description

summary method for objects of class "nlrob", i.e., `nlrob()` results.

Usage

```
## S3 method for class 'nlrob'
summary(object, correlation = FALSE, symbolic.cor = FALSE, ...)
```

Arguments

object	an object of class "nlrob", usually, a result of a call to <code>nlrob</code> .
correlation	logical variable indicating whether to compute the correlation matrix of the estimated coefficients.
symbolic.cor	logical indicating whether to use symbols to display the above correlation matrix.
...	further arguments passed to or from other methods.

Value

The function `summary.nlrob` computes and returns an object of class "summary.nlrob" of summary statistics of the robustly fitted linear model given in object. There is a print method, `print.summary.lmrob()`, which nicely formats the output.

The result keeps a large part of object's components such as residuals, cov or w, and additionally contains

coefficients	the matrix of coefficients, standard errors and p-values.
correlation	if the correlation argument was true, the correlation matrix of the parameters.

Author(s)

Andreas Ruckstuhl

See Also

[nlrob\(\)](#), also for examples.

telef

Number of International Calls from Belgium

Description

Number of international calls from Belgium, taken from the Belgian Statistical Survey, published by the Ministry of Economy.

Usage

```
data(telef)
```

Format

A data frame with 24 observations on the following 2 variables.

calls Number of Calls (in tens of millions)

Year Year (1950 - 1973)

Source

P. J. Rousseeuw and A. M. Leroy (1987) *Robust Regression and Outlier Detection*; Wiley, page 26, table 2.

Examples

```
data(telef)
summary(lm.telef <- lm(Year~., data=telef))
```

tolEllipsePlot	<i>Tolerance Ellipse Plot</i>
----------------	-------------------------------

Description

Plots the 0.975 tolerance ellipse of the bivariate data set x . The ellipse is defined by those data points whose distance is equal to the squareroot of the 0.975 chisquare quantile with 2 degrees of freedom.

Usage

```
tolEllipsePlot(x, m.cov = covMcd(x), cutoff = NULL, id.n = NULL,
              classic = FALSE, tol = 1e-07,
              xlab = "", ylab = "",
              main = "Tolerance ellipse (97.5%)",
              txt.leg = c("robust", "classical"),
              col.leg = c("red", "blue"),
              lty.leg = c("solid", "dashed"))
```

Arguments

<code>x</code>	a two dimensional matrix or data frame.
<code>m.cov</code>	an object similar to those of class "mcd"; however only its components center and cov will be used. If missing, the MCD will be computed (via <code>covMcd()</code>).
<code>cutoff</code>	numeric distance needed to flag data points outside the ellipse.
<code>id.n</code>	number of observations to be identified by a label. If not supplied, the number of observations with distance larger than <code>cutoff</code> is used.
<code>classic</code>	whether to plot the classical distances as well, FALSE by default.
<code>tol</code>	tolerance to be used for computing the inverse, see <code>solve</code> . Defaults to $1e-7$.
<code>xlab</code> , <code>ylab</code> , <code>main</code>	passed to <code>plot.default</code> .
<code>txt.leg</code> , <code>col.leg</code> , <code>lty.leg</code>	character vectors of length 2 for the legend, only used if <code>classic = TRUE</code> .

Author(s)

Peter Filzmoser, Valentin Todorov and Martin Maechler

See Also

`covPlot` which calls `tolEllipsePlot()` when desired. `ellipsoidhull` and `predict.ellipsoid` from package `cluster`.

Examples

```

data(hbk)
hbk.x <- data.matrix(hbk[, 1:3])
mcd <- covMcd(hbk.x)      # compute mcd in advance
## must be a 2-dimensional data set: take the first two columns :
tolEllipsePlot(hbk.x[,1:2])

## an "impressive" example:
data(telef)
tolEllipsePlot(telef, classic=TRUE)

```

toxicity

Toxicity of Carboxylic Acids Data

Description

The aim of the experiment was to predict the toxicity of carboxylic acids on the basis of several molecular descriptors.

Usage

```
data(toxicity)
```

Format

A data frame with 38 observations on the following 10 variables which are attributes for carboxylic acids:

toxicity aquatic toxicity, defined as $\log(IGC_{50}^{-1})$; typically the “response”.

logKow *logKow*, the partition coefficient

pKa pKa: the dissociation constant

ELUMO Energy of the **l**owest **u**noccupied **m**olecular **o**rbital

Ecarb Electrotopological state of the **car**boxylic group

Emet Electrotopological state of the **met**hyl group

RM Molar refractivity

IR Refraction index

Ts Surface tension

P Polarizability

Source

The website accompanying the MMY-book: http://www.wiley.com/legacy/wileychi/robust_statistics

References

Maguna, F.P., Núñez, M.B., Okulik, N.B. and Castro, E.A. (2003) Improved QSAR analysis of the toxicity of aliphatic carboxylic acids; *Russian Journal of General Chemistry* **73**, 1792–1798.

Examples

```
data(toxicity)
summary(toxicity)
plot(toxicity)
plot(toxicity ~ pKa, data = toxicity)

## robustly scale the data (to scale 1) using Qn
(scQ.tox <- sapply(toxicity, Qn))
scTox <- scale(toxicity, center = FALSE, scale = scQ.tox)
csT <- covOGK(scTox, n.iter = 2,
              sigmamu = s_Qn, weight.fn = hard.rejection)
as.dist(round(cov2cor(csT$cov), 2))
```

 tukeyChi

Tukey's "Chi", the Bi-square Loss (Rho) Function

Description

Computes Tukey's bi-square loss function, $\chi(x)$ and its first two derivatives. Note that in the general context of M -estimators, these loss functions are called $\rho(\text{rho})$ -functions.

Usage

```
tukeyChi(x, cc, deriv = 0)
```

Arguments

x	numeric vector.
cc	tuning constant
deriv	integer in $\{0, 1, 2\}$ specifying the order of the derivative; the default, <code>deriv = 0</code> computes the chi- (or rho-)function.

Value

a numeric vector of the same length as x.

Note

`tukeyChi(x, d)` and `tukeyPsi1(x, d-1)` are just re-scaled versions of each other (for d in 0:2).

Author(s)

Matias Salibian-Barrera and Martin Maechler

See Also

[lmrob](#) and [tukeyPsi1](#).

Examples

```
op <- par(mfrow = c(3,1), oma = c(0,0, 2, 0),
          mgp = c(1.5, 0.6, 0), mar = .1+c(3,4,3,2))
x <- seq(-2.5, 2.5, length = 201)
cc <- 1.55 # as set by default in lmrob.control()
plot. <- function(...) { plot(...); abline(h=0,v=0, col="gray", lty=3)}
plot(x, tukeyChi(x, cc), type = "l", col = 2)
plot(x, tukeyChi(x, cc, deriv = 1), type = "l", col = 2)
plot(x, tukeyChi(x, cc, deriv = 2), type = "l", col = 2)

mtext(sprintf("tukeyChi(x, c = %g, deriv), deriv = 0,1,2", cc),
       outer = TRUE, font = par("font.main"), cex = par("cex.main"))
par(op)
```

tukeyPsi1

Tukey's Bi-square Score (Psi) Function and Derivative

Description

Compute Tukey's bi-square score (psi) function, its first derivative or its integral/"principal function". This is scaled such that $\psi'(0) = 1$, i.e., $\psi(x) \approx x$ around 0.

Usage

```
tukeyPsi1(x, cc, deriv = 0)
```

Arguments

x	numeric vector.
cc	tuning constant
deriv	integer in $\{-1, 0, 1\}$ specifying the order of the derivative; the default, <code>deriv = 0</code> computes the psi-function.

Value

a numeric vector of the same length as x.

Note

`tukeyPsi1(x, d)` and `tukeyChi(x, d+1)` are just re-scaled versions of each other (for `d` in $-1:1$), i.e.,

$$\chi^{(\nu)}(x, c) = (6/c^2)\psi^{(\nu-1)}(x, c),$$

for $\nu = 0, 1, 2$.

We use the name 'tukeyPsi1', because `tukeyPsi` is reserved for a future "Psi Function" class object, see [psiFunc](#).

Author(s)

Matias Salibian-Barrera, Martin Maechler and Andreas Ruckstuhl

See Also

[lmrob](#) and [tukeyChi](#); further [anova.lmrob](#) which needs the `deriv = -1`.

Examples

```
op <- par(mfrow = c(3,1), oma = c(0,0, 2, 0),
         mgp = c(1.5, 0.6, 0), mar= .1+c(3,4,1,1))
x <- seq(-5, 5, length = 201)
cc <- 4.69 # as set by default in lmrob.control()
plot.<- function(...) { plot(..., asp = 1); abline(h=0,v=0, col="gray", lty=3)}
plot(x, tukeyPsi1(x, cc), type = "l", col = 2)
abline(0:1, lty = 3, col = "light blue")
plot(x, tukeyPsi1(x, cc, deriv = -1), type = "l", col = 2)
plot(x, tukeyPsi1(x, cc, deriv = 1), type = "l", col = 2); abline(h=1,lty=3)

mtext(sprintf("tukeyPsi1(x, c = %g, deriv),  deriv = 0, -1, 1", cc),
       outer = TRUE, font = par("font.main"), cex = par("cex.main"))
par(op)
```

vaso

Vaso Constriction Data Set

Description

Finney's data on vaso constriction in the skin of the digits.

Usage

```
data(vaso)
```

Format

A data frame with 39 observations on the following 3 variables.

Volume Inhaled volume of air

Rate Rate of inhalation

Y vector of 0 or 1 values.

Details

The data taken from Finney (1947) were obtained in a carefully controlled study in human physiology where a reflex “vaso constriction” may occur in the skin of the digits after taking a single deep breath. The response y is the occurrence ($y = 1$) or non-occurrence ($y = 0$) of vaso constriction in the skin of the digits of a subject after he or she inhaled a certain volume of air at a certain rate. The responses of three subjects are available. The first contributed 9 responses, the second contributed 8 responses, and the third contributed 22 responses.

Although the data represent repeated measurements, an analysis that assumes independent observations may be applied, as claimed by Pregibon (1981).

Source

Finney, D.J. (1947) The estimation from individual records of the relationship between dose and quantal response. *Biometrika* **34**, 320–334

References

Atkinson, A.C. and Riani, M. (2000) *Robust Diagnostic Regression Analysis*, First Edition. New York: Springer, Table A.23.

Fahrmeir, L. and Tutz, G. (2001) *Multivariate Statistical Modelling Based on Generalized Linear Models*, Springer, Table 4.2.

Kuensch, H.R., Stefanski, A. and Carroll, R.J. (1989) Conditionally unbiased bounded influence estimation in general regression models, with applications to generalized linear models, *JASA* **84**, 460–466.

Pregibon, D. (1981) Logistic regression diagnostics, *Annals of Statistics* **9**, 705–724.

Examples

```
data(vaso)
str(vaso)
pairs(vaso)

glmV <- glm(Y ~ log(Volume) + log(Rate), family=binomial, data=vaso)
summary(glmV)
## --> example(glmrob) showing classical & robust GLM
```

wagnerGrowth

Wagner's Hannover Employment Growth Data

Description

Wagner (1994) investigates the rate of employment growth (y) as function of percentage of people engaged in **production activities** (PA) and **higher services** (HS) and of the **growth** of these percentages (GPA, GHS) during three time periods in 21 geographical regions of the greater Hannover area.

Usage

```
data(wagnerGrowth)
```

Format

A data frame with $21 \times 3 = 63$ observations (one per Region \times Period) on the following 7 variables.

Region a **factor** with 21 levels, denoting the corresponding region in Hannover (conceptually a “block factor”).

PA numeric: percent of people involved in production activities.

GPA **g**rowth of PA.

HS a numeric vector

GHS a numeric vector

y a numeric vector

Period a **factor** with levels 1:3, denoting the time period, 1 = 1979-1982, 2 = 1983-1988, 3 = 1989-1992.

Source

Hubert, M. and Rousseeuw, P. J. (1997). Robust regression with both continuous and binary regressors, *Journal of Statistical Planning and Inference* **57**, 153–163.

References

Wagner J. (1994). Regionale Beschäftigungsdynamik und höherwertige Produktionsdienste: Ergebnisse für den Grossraum Hannover (1979-1992). *Raumforschung und Raumordnung* **52**, 146–150.

Examples

```
data(wagnerGrowth)
## maybe
str(wagnerGrowth)
```

```
require(lattice)
(xyplot(y ~ Period | Region, data = wagnerGrowth,
        main = "wagnerGrowth: 21 regions @ Hannover"))
```

```
(dotplot(y ~ reorder(Region,y,median), data = wagnerGrowth,
          main = "wagnerGrowth",
          xlab = "Region [ordered by median(y | Region) ]"))
```

wgt.himedian	<i>Weighted Hi-Median</i>
--------------	---------------------------

Description

Compute the weighted Hi-Median of x .

Usage

```
wgt.himedian(x, weights = rep(1, n))
```

Arguments

x	numeric vector
weights	numeric vector of weights; of the same length as x .

Note

this is rather a by-product of the code used in [Sn](#) and [Qn](#). We currently plan to replace it with more general weighted quantiles.

See Also

[median](#); also [wtd.quantile](#) from package **Hmisc**.

Examples

```
x <- c(1:6, 20)
median(x) ## 4
stopifnot(all.equal(4, wgt.himedian(x)),
           all.equal(6, wgt.himedian(x, c(rep(1,6), 5))))
```

wood	<i>Modified Data on Wood Specific Gravity</i>
------	---

Description

The original data are from Draper and Smith (1966) and were used to determine the influence of anatomical factors on wood specific gravity, with five explanatory variables and an intercept. These data were contaminated by replacing a few observations with outliers.

Usage

```
data(wood)
```

Format

A data frame with 20 observations on the following 6 variables.

x1, x2, x3, x4, x5 explanatory “anatomical” wood variables.

y wood specific gravity, the target variable.

Source

Draper and Smith (1966, p.227)

Peter J. Rousseeuw and Annick M. Leroy (1987) *Robust Regression and Outlier Detection* Wiley, p.243, table 8.

Examples

```
data(wood)
plot(wood)
summary(lm.wood <- lm(y ~ ., data = wood))
summary(rlm.wood <- MASS::rlm(y ~ ., data = wood))
summary(lts.wood <- ltsReg(y ~ ., data = wood))

wood.x <- as.matrix(wood)[,1:5]
c_wood <- covMcd(wood.x)
c_wood
```

Index

*Topic **L1**

lmrob.lar, 61

*Topic **M-S**

lmrob.M.S, 62

*Topic **arith**

h.alpha.n, 44

*Topic **classes**

functionX-class, 37

functionXal-class, 38

psi_func-class, 92

psiFunc, 91

*Topic **datasets**

aircraft, 10

airmay, 11

alcohol, 12

ambientNOxCH, 13

Animals2, 16

bushfire, 21

carrots, 22

cloud, 23

coleman, 24

condroz, 25

CrohnD, 31

cushny, 32

delivery, 33

education, 34

epilepsy, 35

exAM, 36

hbk, 45

heart, 46

kootenay, 48

lactic, 49

los, 66

milk, 72

NOxEmissions, 76

pension, 78

phosphor, 79

pilot, 79

possumDiv, 85

pulpfiber, 93

radarImage, 95

salinity, 99

SiegelsEx, 101

starsCYG, 105

telef, 113

toxicity, 115

vaso, 118

wagnerGrowth, 119

wood, 121

*Topic **hplot**

adjbox, 4

plot.lts, 81

plot.mcd, 83

tolEllipsePlot, 114

*Topic **methods**

chgDefaults-methods, 23

*Topic **models**

anova.glmrob, 17

anova.lmrob, 19

predict.glmrob, 87

residuals.glmrob, 96

*Topic **multivariate**

adjOutlyingness, 8

covMcd, 26

covOGK, 28

plot.lts, 81

plot.mcd, 83

rrcov.control, 97

summary.mcd, 111

*Topic **nonlinear**

glmrob, 38

glmrobMqle.control, 43

nlrob, 73

summary.glmrob, 107

summary.nlrob, 112

*Topic **regression**

anova.glmrob, 17

anova.lmrob, 19

- glmrob, 38
- glmrobMqle.control, 43
- lmrob, 50
- lmrob..D..fit, 53
- lmrob..M..fit, 55
- lmrob.control, 57
- lmrob.fit, 60
- lmrob.lar, 61
- lmrob.M.S, 62
- lmrob.S, 64
- ltsReg, 67
- nlrob, 73
- plot.lmrob, 80
- predict.glmrob, 87
- predict.lmrob, 89
- print.lmrob, 90
- residuals.glmrob, 96
- summary.glmrob, 107
- summary.lmrob, 108
- summary.lts, 109
- summary.nlrob, 112
- *Topic robust**
 - adjboxStats, 7
 - adjOutlyingness, 8
 - anova.glmrob, 17
 - anova.lmrob, 19
 - covMcd, 26
 - covOGK, 28
 - exAM, 36
 - glmrob, 38
 - glmrobMqle.control, 43
 - huberM, 47
 - lmrob, 50
 - lmrob..D..fit, 53
 - lmrob..M..fit, 55
 - lmrob.control, 57
 - lmrob.fit, 60
 - lmrob.M.S, 62
 - lmrob.S, 64
 - ltsReg, 67
 - mc, 70
 - nlrob, 73
 - plot.lmrob, 80
 - plot.mcd, 83
 - predict.lmrob, 89
 - print.lmrob, 90
 - psi_func-class, 92
 - psiFunc, 91
 - Qn, 94
 - rrcov.control, 97
 - scaleTau2, 100
 - Sn, 102
 - summary.glmrob, 107
 - summary.lmrob, 108
 - summary.lts, 109
 - summary.mcd, 111
 - summary.nlrob, 112
 - tolEllipsePlot, 114
 - tukeyChi, 116
 - tukeyPsi1, 117
 - wgt.himedian, 121
- *Topic univar**
 - adjboxStats, 7
 - huberM, 47
 - mc, 70
 - Qn, 94
 - scaleTau2, 100
 - Sn, 102
 - wgt.himedian, 121
- *Topic utilities**
 - summarizeRobWeights, 106
- .Random.seed, 57, 98
- adjbox, 4, 7, 8, 10
- adjboxStats, 4, 7
- adjOutlyingness, 8
- aircraft, 10
- airmay, 11
- alcohol, 12
- ambientNOxCH, 13, 77
- Animals, 16
- Animals2, 16
- anova, 18, 20
- anova.glmrob, 17, 97
- anova.lmrob, 19, 20, 118
- as.data.frame, 50
- boxplot, 6
- boxplot.default, 6
- boxplot.stats, 5–8
- bushfire, 21
- bxp, 4, 5
- carrots, 22
- chgDefaults (chgDefaults-methods), 23
- chgDefaults, ANY-method
 - (chgDefaults-methods), 23

- chgDefaults,psi_func-method
(chgDefaults-methods), 23
- chgDefaults-methods, 23
- class, 107, 111
- cloud, 23
- coef, 70, 97
- coefficients, 40, 69, 75
- coleman, 24
- condroz, 25
- cov.mcd, 26, 28
- cov.rob, 30, 39
- covGK (covOGK), 28
- covMcd, 26, 30, 39, 44, 58, 70, 81, 84, 98, 111, 112, 114
- covOGK, 28, 28, 95, 101, 102
- covPlot, 83, 114
- covPlot (plot.mcd), 83
- CrohnD, 31
- cushny, 32

- data.frame, 9, 18, 20
- delivery, 33
- dev.interactive, 82, 84

- education, 34
- ellipsoidhull, 114
- epilepsy, 35
- exAM, 36

- factor, 5, 58, 63, 86, 120
- family, 38–40
- fitted, 70, 97, 108
- fitted.nlrob (nlrob), 73
- fitted.values, 69
- formula, 39, 50, 67, 68, 73
- function, 91
- functionX-class, 37
- functionXal-class, 38

- glm, 39, 40, 97
- glmrob, 17, 18, 22, 38, 43, 87, 88, 96, 97, 106–108
- glmrobMqle.control, 39, 41, 43

- h.alpha.n, 26, 27, 44, 68, 69
- hard.rejection (covOGK), 28
- hbk, 45
- heart, 46
- huber, 47
- huberM, 47
- huberPsi (psiFunc), 91
- hubers, 48

- invisible, 112
- IQR, 9, 30

- kootenay, 48

- lactic, 49
- length, 27, 69
- list, 6, 8, 27, 51, 60, 69
- lm, 19, 39, 50
- lmRob, 63
- lmrob, 19, 20, 50, 54, 56–61, 63, 65, 69, 70, 81, 90, 106, 108, 109, 117, 118
- lmrob..D..fit, 53, 61
- lmrob..M..fit, 55, 61
- lmrob.control, 51, 52, 54, 55, 57, 60–62, 64, 98, 103
- lmrob.fit, 51, 52, 54–56, 60, 62–64
- lmRob.lar, 61
- lmrob.lar, 61
- lmrob.M.S, 52, 62, 104
- lmrob.S, 51, 52, 58, 60, 61, 64, 70
- logical, 67
- los, 66
- ltsPlot (plot.lts), 81
- ltsReg, 44, 67, 110, 111

- mad, 30, 47, 48, 95, 100, 101, 103
- mahalanobis, 98
- mammals, 16
- match.call, 27
- matrix, 9
- mc, 5–8, 10, 70
- median, 94, 102, 121
- methods, 97
- milk, 72
- model.frame, 62, 103
- model.matrix, 103, 108
- model.matrix.default, 51, 67
- model.matrix.lmrob (summary.lmrob), 108

- NA, 4, 27, 29, 71
- na.exclude, 51, 67
- na.fail, 51, 67
- na.omit, 39, 51, 67, 77
- nlrob, 73, 112, 113

- nls, 73–75
- nls.control, 74
- NOxEmissions, 14, 76
- offset, 51, 68
- options, 39, 51, 67, 90, 107, 109
- par, 82, 84
- pension, 78
- phosphor, 79
- pilot, 79
- plot, 69
- plot.default, 114
- plot.lm, 81
- plot.lmrob, 52, 80
- plot.lts, 81
- plot.mcd, 83
- possum.mat (possumDiv), 85
- possumDiv, 85
- predict, 108
- predict.ellipsoid, 114
- predict.glmrob, 41, 87
- predict.lm, 88, 90
- predict.lmrob, 52, 88, 89, 108, 109
- predict.nlrob (nlrob), 73
- predict.nls, 74
- print, 90, 106–109, 111
- print.default, 111
- print.lmrob, 52, 90
- print.lts (ltsReg), 67
- print.mcd, 111
- print.mcd (covMcd), 26
- print.summary.glmrob (summary.glmrob), 107
- print.summary.lmrob (summary.lmrob), 108
- print.summary.lts (summary.lts), 109
- print.summary.mcd (summary.mcd), 111
- printCoefmat, 110
- psi_func-class, 92
- psiFunc, 37, 38, 91, 92, 117
- pulpfiber, 93
- Qn, 71, 94, 101, 103, 121
- qr, 9, 68
- radarImage, 95
- residuals, 69, 70, 97, 108
- residuals.glm, 97
- residuals.glmrob, 40, 88, 96
- residuals.nlrob (nlrob), 73
- rlm, 36, 56, 74, 75
- rnls, 75
- rq, 62
- rrcov.control, 26, 68, 97
- runif, 57
- s_IQR (covOGK), 28
- s_mad (covOGK), 28
- s_Qn, 29
- s_Qn (Qn), 94
- s_Sn, 29
- s_Sn (Sn), 102
- salinity, 99
- scaleTau2, 29, 30, 95, 100, 103
- SiegelsEx, 101
- sleep, 32
- Sn, 95, 101, 102, 121
- solve, 84, 98, 114
- solve.qr, 9
- splitFrame, 58, 62, 63, 103
- starsCYG, 105
- summarizeRobWeights, 106
- summary, 40, 69, 75, 90, 106, 108, 109, 111, 112
- summary.glm, 108
- summary.glmrob, 40, 97, 107, 107
- summary.lm, 109
- summary.lmrob, 52, 90, 108
- summary.lts, 70, 109
- summary.mcd, 111
- summary.nlrob, 112, 112
- symnum, 107
- telef, 113
- terms, 40
- text, 84
- tolEllipsePlot, 84, 85, 114
- toxicity, 115
- tukeyChi, 116, 117, 118
- tukeyPsi1, 116, 117, 117
- vaso, 118
- vcov, 108
- vcov.glmrob (summary.glmrob), 107
- vcov.lmrob (summary.lmrob), 108
- wagnerGrowth, 119
- weights, 108

wgt.himedian, 47, 121

wood, 121

wtd.quantile, 121