

The robust Package

March 9, 2007

Version 0.3-0

Date 2007-03-09

Title Insightful Robust Library

Author Jeff Wang <jwang@statsci.com>, Ruben Zamar <ruben@stat.ubc.ca>, Alfio Marazzi <Alfio.Marazzi@inst.hospvd.ch>, Victor Yohai <vyohai@dm.uba.ar>, Matias Salibian-Barrera <matias@stat.ubc.ca>, Ricardo Maronna <maron@mate.unlp.edu.ar>, Eric Zivot <ezivot@u.washington.edu>, David Rocke <dmrocke@ucdavis.edu>, Doug Martin <doug@statsci.com>, Kjell Konis <konis@stats.ox.ac.uk>.

Maintainer Kjell Konis <konis@stats.ox.ac.uk>

Depends R (>= 2.4.1), MASS, lattice

Description A package of robust methods.

License GPL

R topics documented:

.First.lib	3
add1.lmRob	4
anova.glmRob	5
anova.lmRob	6
arg.names	7
breslow.dat	7
coef.glmfm	8
coef.lmfm	9
cov	10
covRob	11
covRob.control	12
covfmDistance2Plot	13
covfmEllipsesPlot	14
covfmScreePlot	15
covfmSqrtMDPlot	16
donostah	17

drop1.lmRob	18
fastcov	19
fastmcd	21
fit.models	23
gen.data	25
get.fit.models.database	25
glmRob.Initial.LMS	26
glmRob	26
glmRob.control	28
glmRob.cubif.Ini	29
glmRob.cubif	29
glmRob.cubif.control	30
glmRob.cubif.null	31
glmRob.mallows	32
glmRob.mallows.control	33
glmRob.misclass	33
glmRob.misclass.control	34
glmRob.misclass.w	35
glmRob.object	35
key	37
leuk.dat	38
lmRob.RFPE	38
lmRob	39
lmRob.control	41
lmRob.effvy	42
lmRob.fit.compute	43
lmRob.ga	44
lmRob.object	44
lmRob.ucovcoef	46
lmfm2DRegPlot	46
lmfmOverlaidQQPlot	47
lmfmOverlaidResDenPlot	47
lmfmResKernDenPlot	48
lmfmResQQPlot	48
lmfmResVsFittedPlot	49
lmfmRespVsFittedPlot	50
lmfmSRvsRDPlot	50
lmfmSqrtResVsFittedPlot	51
lmfmStdResPlot	52
mallows.dat	52
methods.glmRob	53
methods.lmRob	54
panel.addons	55
plot.covfm	56
plot.glmRob	57
plot.lmfm	58
plot.lmRob	59
plot.lmfm	60

predict.glmRob 61

predict.lmRob 63

print.covfm 64

print.glmRob 65

print.glmfm 65

print.lmRob 66

print.lmfm 67

print.summary.covfm 68

print.summary.glmRob 69

print.summary.glmfm 70

print.summary.lmRob 70

print.summary.lmfm 71

qqplot.glmRob 72

rb 73

rockem 73

s.logistic.misclass.fit 75

splus.assign 76

stack.dat 76

step.lmRob 77

stop.on.bdObject 78

summary.covfm 79

summary.glmRob 80

summary.glmfm 81

summary.lmRob 81

summary.lmfm 82

test.lmRob 83

undocumented.glmRob 84

undocumented.lmRob 86

update.lmRob 87

weight.funs.q 88

woodmod.dat 89

wt.carroll 90

Index **91**

.First.lib *First Run*

Description

Loads native code and initializes .Random.seed.

Usage

.First.lib(libname, pkgname)

Arguments

libname	library
pkgname	section

Value

a single uniform random number is returned invisibly.

 add1.lmRob

Add Terms to a Robust Linear Model Object

Description

add1.lmRob is used to investigate an lmRob object by adding to it, in turn, each of a number of specified terms.

Usage

```
## S3 method for class 'lmRob':
add1(object, scope=. ~ ., scale, keep, ...)
```

Arguments

object	an lmRob object.
scope	a formula object describing the terms to be added. This argument is required, and is parsed to produce a set of terms that may be added to the model on their own without breaking the hierarchy rules. The scope can also be a character vector of term labels. Any "." in scope is interpreted relative to the formula implied by the object argument.
scale	an estimate of the scale of the residuals. If not supplied, the initial estimate of the scale in object is used.
keep	a character vector of names of components that should be saved for each augmented model. Only names from the set "coefficients", "fitted" and "residuals" are allowed. If keep is TRUE, the complete set is implied. The default behavior is not to keep anything.
...	additional arguments required by the generic add1 function.

Details

An anova object is constructed, consisting of the term labels, the degrees of freedom, and Robust Final Prediction Errors (RFPE) for each superset model.

This function implements the generic function [add1](#) for lmRob class objects.

Value

if `keep` is missing, an `anova` object corresponding to each superset model implied by `object` and `scope`; otherwise, a list with components:

<code>anova</code>	an <code>anova</code> object corresponding to each superset model implied by <code>object</code> and <code>scope</code> .
<code>keep</code>	a matrix of mode <code>"list"</code> , with a column for each superset model, and a row for each component kept.
<code>...</code>	additional arguments required by the generic <code>add1</code> function.

See Also

[lmRob](#), [add1](#), [anova](#), [drop1](#), [lmRob.object](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ Water.Temp, data = stack.dat)
add1(stack.rob, . ~ . + Air.Flow + Acid.Conc.)
```

anova.glmRob

ANOVA for Robust Generalized Linear Model Fits

Description

Compute an analysis of variance table for one or more robust generalized linear model fits.

Usage

```
## S3 method for class 'glmRob':
anova(object, ..., test = c("none", "Chisq", "F", "Cp"))
anova.glmRoblist(object, ..., test = c("none", "Chisq", "F", "Cp"))
```

Arguments

<code>object</code>	a <code>glmRob</code> object.
<code>...</code>	additional <code>glmRob</code> objects.
<code>test</code>	a character string specifying the test statistic to be used. Can be one of <code>"F"</code> , <code>"Chisq"</code> , <code>"Cp"</code> or <code>"none"</code> for no test.

Value

an `anova` object.

See Also

[glmRob](#), [anova](#), [anova.glmRoblist](#).

Examples

```
data(breslow.dat)
bres.int <- glmRob(sumY ~ Age10 + Base4*Trt, family = poisson(), data = breslow.dat)
bres.main <- glmRob(sumY ~ Age10 + Base4 + Trt, family = poisson(), data = breslow.dat)
anova(bres.int)
anova(bres.int, bres.main)
```

anova.lmRob

ANOVA for Robust Linear Model Fits

Description

Compute an analysis of variance table for one or more robust linear model fits.

Usage

```
## S3 method for class 'lmRob':
anova(object, ..., test = c("RF", "RWald"))
anova.lmRoblist(object, const, ipsi, yc, test = c("RWald", "RF"), ...)
```

Arguments

<code>object</code>	an <code>lmRob</code> object.
<code>...</code>	additional arguments required by the generic <code>anova</code> function. If <code>...</code> contains additional robustly fitted linear models then the function <code>anova.lmRoblist</code> is dispatched.
<code>const</code>	a numeric value containing the tuning constant computed by <code>lmRob.const</code> .
<code>ipsi</code>	an integer value specifying the psi-function.
<code>yc</code>	a numeric value containing the tuning constant computed by <code>lmRob.effvy</code> .
<code>test</code>	a single character value specifying which test should be computed in the Anova table. The possible choices are "RWald" and "RF".

Details

The default test used by `anova` is the "RWald" test, which is the Wald test based on robust estimates of the coefficients and covariance matrix. If `test` is "RF", the robustified F-test is used instead.

Value

an `anova` object.

References

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. (1986). *Robust statistics: the approach based on influence functions*. John Wiley & Sons.

See Also

[lmRob](#), [anova](#).

Examples

```
data(stack.dat)
stack.small <- lmRob(Loss ~ Water.Temp + Acid.Conc., data = stack.dat)
stack.full <- lmRob(Loss ~ ., data = stack.dat)
anova(stack.full)
anova(stack.full, stack.small)
```

arg.names

Argument Names

Description

Returns the argument names of a function or call.

Usage

```
arg.names(x)
```

Arguments

`x` a function, call, or a character vector of length 1 containing the name of a function.

Value

a character vector containing the names of the arguments.

Examples

```
arg.names(lm)
```

`breslow.dat`*Breslow Data*

Description

Patients suffering from simple or complex partial seizures were randomized to receive either the antiepileptic drug progabide or a placebo. At each of four successive postrandomization clinic visits, the number of seizures occurring over the previous two weeks was reported.

Usage

```
data(breslow.dat)
```

Format

A data frame with 59 observations on the following 12 variables.

ID an integer value specifying the patient identification number.

Y1 an integer value, the number of seizures during the first two week period.

Y2 an integer value, the number of seizures during the second two week period.

Y3 an integer value, the number of seizures during the third two week period.

Y4 an integer value, the number of seizures during the fourth two week period.

Base an integer value giving the eight-week baseline seizure count.

Age an integer value giving the age of the patient in years.

Trt the treatment: a factor with levels `placebo` and `progabide`.

Ysum an integer value, the sum of Y1, Y2, Y3 and Y4.

sumY an integer value, the sum of Y1, Y2, Y3 and Y4.

Age10 a numeric value, Age divided by 10.

Base4 a numeric value, Base divided by 4.

References

Breslow, N. E., and Clayton, D. G. (1993), "Approximate Inference in Generalized Linear Mixed Models," *Journal of the American Statistical Association*, Vol. 88, No. 421, pp. 9-25.

Thrall, P. F., and Vail, S. C. (1990), "Some Covariance Models for Longitudinal Count Data With Overdispersion," *Biometrics*, Vol. 46, pp. 657-671.

Examples

```
data(breslow.dat)
```

coef.glmfm	<i>Extract Model Coefficients</i>
------------	-----------------------------------

Description

Extract the model coefficients from a glmfm object.

Usage

```
## S3 method for class 'glmfm':  
coef(object, ...)
```

Arguments

object a glmfm object.
... additional arguments required by the generic `coef` function.

Value

a numeric matrix with one row for each model in `object` and one column for each coefficient in the model.

Examples

```
data(breslow.dat)  
bres.fm <- fit.models(list(Robust = "glmRob", Classical = "glm"),  
                      formula = sumY ~ Age10 + Base4*Trt,  
                      family = poisson(), data = breslow.dat)  
  
coef(bres.fm)
```

coef.lmfmm	<i>Extract Model Coefficients</i>
------------	-----------------------------------

Description

Extract the model coefficients from an lmfmm object.

Usage

```
## S3 method for class 'lmfmm':  
coef(object, ...)
```

Arguments

object an lmfmm object.
... additional arguments required by the generic `coef` function.

Value

a numeric matrix with one row for each model in `object` and one column for each coefficient in the model.

Examples

```
data(stack.dat)
stack.fm <- fit.models(list(Robust = "lmRob", LS = "lm"), Loss ~ ., data = stack.dat)
coef(stack.fm)
```

 cov

Classical Covariance Estimation

Description

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

Usage

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

Arguments

<code>data</code>	a numeric matrix or data frame containing the data.
<code>corr</code>	a logical flag. If <code>corr = TRUE</code> then the estimated correlation matrix is computed.
<code>center</code>	a logical flag or a numeric vector of length <code>p</code> (where <code>p</code> is the number of columns of <code>x</code>) specifying the center. If <code>center = TRUE</code> then the center is estimated. Otherwise the center is taken to be 0.
<code>distance</code>	a logical flag. If <code>distance = TRUE</code> the Mahalanobis distances are computed.
<code>na.action</code>	a function to filter missing data. The default <code>na.fail</code> produces an error if missing values are present. An alternative is <code>na.omit</code> which deletes observations that contain one or more missing values.
<code>unbiased</code>	a logical flag. If <code>unbiased = TRUE</code> then an unbiased estimate of the covariance matrix is computed. If <code>unbiased = FALSE</code> then a maximum likelihood estimate is computed.

Details

This function is intended to produce an object similar to that produced by the `covRob` in the `robust` library but fit using classical methods.

Value

an object of class "cov" with components:

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

See Also

[covRob](#), [var](#), [cov.wt](#).

Examples

```
data(stack.dat)
cov(stack.dat)
```

covRob

Robust Covariance/Correlation Matrix Estimation

Description

Computes robust estimates of multivariate location and scatter.

Usage

```
covRob(data, corr = FALSE, distance = TRUE, na.action = na.fail, estim = "auto",
```

Arguments

data	a numeric matrix or data frame containing the data.
corr	a logical flag. If <code>corr = TRUE</code> then the estimated correlation matrix is computed.
distance	a logical flag. If <code>distance = TRUE</code> the Mahalanobis distances are computed.
na.action	a function to filter missing data. The default <code>na.fail</code> produces an error if missing values are present. An alternative is <code>na.omit</code> which deletes observations that contain one or more missing values.

<code>estim</code>	the robust estimator to be used. The choices are: "mcd" for the Fast MCD algorithm of Rousseeuw and Van Driessen, "donostah" for the Donoho-Stahel projection based estimator, "M" for the constrained M estimator provided by Rocke, "pairwiseQC" for the orthogonalized quadrant correlation pairwise estimator, and "pairwiseGK" for the Orthogonalized Gnanadesikan-Kettenring pairwise estimator. The default "auto" selects from "donostah", "mcd", and "pairwiseQC" with the goal of producing a good estimate in a reasonable amount of time.
<code>control</code>	a list of control parameters to be used in the numerical algorithms. See <code>covRob.control</code> for the possible control parameters and their default settings. This argument is ignored when <code>estim = "auto"</code> .
<code>...</code>	control parameters may be passed directly when <code>estim != "auto"</code> .

Details

Value

an object of class "covRob" with components:

<code>call</code>	an image of the call that produced the object with all the arguments named.
<code>cov</code>	a numeric matrix containing the final robust estimate of the covariance/correlation matrix.
<code>center</code>	a numeric vector containing the final robust estimate of the location vector.
<code>raw.cov</code>	a numeric matrix containing the initial robust estimate of the covariance/correlation matrix.
<code>raw.center</code>	a numeric vector containing the initial robust estimate of the location vector.
<code>dist</code>	a numeric vector containing the Mahalanobis distances computed using robust estimates of covariance and location contained in <code>cov</code> and <code>center</code> . Only present if <code>distance = TRUE</code> in the call.
<code>corr</code>	a logical flag. If <code>corr = TRUE</code> then <code>cov</code> and <code>raw.cov</code> contain robust estimates of the correlation matrix of <code>x</code> .
<code>estim</code>	a character vector of length 1 containing the name of the robust estimator.
<code>control</code>	a list containing the control parameters used by the robust estimator.

See Also

[covRob.control](#), [cov](#), [fastmcd](#), [donostah](#), [fastcov](#), [rockem](#).

Examples

```
data(stack.dat)
covRob(stack.dat, estim = "mcd", quan = .75, ntrial = 1000)
```

covRob.control *Control Parameters for Robust Covariance Estimation*

Description

This function is used to create a list of control parameters for the underlying robust estimator used in the `covRob` function.

Usage

```
covRob.control(estim, ...)
```

Arguments

<code>estim</code>	a character vector of length one giving the name of the estimator to generate the control parameters for.
<code>...</code>	control parameters appropriate for the robust estimator specified in <code>estim</code> in the form <code>name = value</code> and separated by commas. Omitted parameters receive their default values.

Details

The control parameters are estimator specific. Information on the control parameters (and their default values) can be found in the help files of each of the robust covariance estimators.

Value

a list of control parameters appropriate for the robust estimator given in `estim`. The value of `estim` occupies the first element of the list.

See Also

This function is a utility function for `covRob`.

The current supported robust estimators are: `donostah`, `fastmcd`, `rockem`, `fastcov`.

Examples

```
mcd.control <- covRob.control("mcd", quan = 0.75, ntrial = 1000)
ds.control <- covRob.control("donostah", prob = 0.95)
qc.control <- covRob.control("pairwiseqc")
```

covfmDistance2Plot *Distance - Distance Plot*

Description

Given a `fit.models` object containing two elements of either class "cov" or "covRob", plots the square root of the Mahalanobis distances from the first element on the y-axis and the square root of the Mahalanobis distances from the second element on the x-axis. A 45 degree line is drawn as well.

Usage

```
covfmDistance2Plot(x, chisq.percent = 0.975, id.n = 3, ...)
```

Arguments

`x` an object of class "covfm" containing 2 elements.

`chisq.percent` a numeric value between 0 and 1 giving the chi-squared percent point used to compute the outlyingness threshold.

`id.n` a positive integer specifying the number of extreme points to label in the plot.

`...` additional arguments are passed to the low-level plotting functions.

Details

This function is called by the generic function `plot.covfm`.

Value

`x` is invisibly returned.

See Also

`plot.covfm`, `fit.models`, `covRob`, `cov`.

Examples

```
data(woodmod.dat)
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
covfmDistance2Plot(woodmod.fm)
```

`covfmEllipsesPlot` *Ellipses Plot*

Description

When there are 2 variables in the data this function produces a scatter plot of the data with an overlaid ellipse for each model in the covfm object. When there are more than 2 variables this function draws a matrix with ellipses in the upper triangle. The ellipse in the i,j cell of the matrix is drawn to be a contour of a standard bivariate normal with correlation equal to the i, j element of the correlation matrix. There is one ellipse in each cell for each model in the covfm object.

Usage

```
covfmEllipsesPlot(x, ...)
```

Arguments

`x` an object of class "covfm" containing 2 elements.
`...` additional arguments are passed to the low-level plotting functions.

Details

This function is called by the generic function `plot.covfm`. To use this function on a "cov" or "covRob" object first use `fit.models` to coerce the object to class "covfm".

Value

`x` is invisibly returned.

See Also

`plot.covfm`, `fit.models`, `covRob`, `cov`.

Examples

```
data(woodmod.dat)
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
covfmEllipsesPlot(woodmod.fm)
```

covfmScreePlot *Screeplot*

Description

Draws overlaid screeplots for the models in a covfm object.

Usage

```
covfmScreePlot(x, variables, ...)
```

Arguments

`x` an object of class "covfm" containing 2 elements.

`variables` a vector of positive integers specifying which eigenvalues are to be drawn. By default the 10 largest eigenvalues are plotted.

`...` additional arguments are passed to the low-level plotting functions.

Details

This function is called by the generic function `plot.covfm`. To use this function on a "cov" or "covRob" object first use `fit.models` to coerce the object to class "covfm".

Value

`x` is invisibly returned.

See Also

`plot.covfm`, `fit.models`, `covRob`, `cov`.

Examples

```
data(woodmod.dat)
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
covfmScreePlot(woodmod.fm)
```

covfmSqrtMDPlot *Square Root of Mahalanobis Distance Plot*

Description

Produces side-by-side plots of the Mahalanobis distances computed using the location and covariance matrix estimates contained in each model of a covfm object.

Usage

```
covfmSqrtMDPlot(x, chisq.percent = 0.975, id.n = 3, ...)
```

Arguments

`x` an object of class "covfm" containing 2 elements.
`chisq.percent` a numeric value between 0 and 1 giving the chi-squared percent point used to compute the outlyingness threshold.
`id.n` a positive integer specifying the number of extreme points to label in the plot.
`...` additional arguments are passed to the low-level plotting functions.

Details

This function is called by the generic function `plot.covfm`. To use this function on a "cov" or "covRob" object first use `fit.models` to coerce the object to class "covfm".

Value

`x` is invisibly returned.

See Also

`plot.covfm`, `fit.models`, `covRob`, `cov`.

Examples

```
data(woodmod.dat)
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
covfmSqrtMDPlot(woodmod.fm)
```

 donostah

Donoho-Stahel Covariance Estimation

Description

Compute a robust estimate of location and scale using the Donoho-Stahel projection-based estimator.

Usage

```
donostah(x, control)
```

Arguments

x a numeric matrix containing the data.

control a list of control parameters. The utility function `covRob.control` creates a list of the control parameters and their default values. See details for the required control parameters and their default values.

Details

This function is called by the high-level function `covRob` when the Stahel-Donoho estimator is specified (via the optional argument `estim = "donostah"`). It may also be of interest to power users who want to compute a Stahel-Donoho estimate with a minimum of fuss.

nresamp = "auto" a positive integer giving the number of resamples required; `nresamp` may not be reached if too many of the size `p` subsamples, chosen out of the observed vectors, are in a hyperplane. If `nresamp = 0` all subsamples are taken. The default `nresamp = "auto"` is calculated to provide a breakdown point of `eps` with probability `prob`.

maxres = "auto" a positive integer specifying the maximum number of resamples to be performed including those that are discarded due to linearly dependent subsamples. The default value `maxres = "auto"` is 2 times `nresamp`.

random.sample = FALSE a logical flag. If `FALSE` then the random seed is set so that the estimate is reproducible.

center = TRUE a logical flag or numeric vector containing the location about which the covariance is to be taken. If `center = TRUE` then a robust estimate of the center is computed; if `center = FALSE` then no centering takes place and the center is taken to be the zero vector.

tune = 0.95 a numeric value between 0 and 1 giving the fraction of the data to receive non-zero weight.

prob = 0.99 a numeric value between 0 and 1 specifying the probability of high breakdown point; used to compute `nresamp` when `nresamp = "auto"`.

eps = 0.5 a numeric value between 0 and 0.5 specifying the breakdown point; used to compute `nresamp` when `nresamp = "auto"`.

Value

a list with the following components:

<code>call</code>	an image of the call that produced the object with all the arguments named.
<code>cov</code>	a numeric matrix containing the Stahel-Donoho estimate of the covariance/correlation matrix.
<code>center</code>	a numeric vector containing the Stahel-Donoho estimate of the location vector.

References

Maronna, R. A. and Yohai, V. J. (1995). The behavior of the Stahel-Donoho robust multivariate estimator. *Journal of the American Statistical Association*, 90, 330-341.

See Also

`covRob`, `covRob.control`.

Examples

```
data(woodmod.dat)
X <- as.matrix(woodmod.dat)
ds.control <- covRob.control("donostah")
donostah(X, ds.control)
```

`drop1.lmRob`

Compute an Anova Object by Dropping Terms

Description

`drop1.lmRob` is used to investigate a robust Linear Model object by recomputing it, successively omitting each of a number of specified terms.

Usage

```
## S3 method for class 'lmRob':
drop1(object, scope, scale, keep, fast = FALSE, ...)
```

Arguments

<code>object</code>	an <code>lmRob</code> object.
<code>scope</code>	an optional formula object describing the terms to be dropped. Typically this argument is omitted, in which case all possible terms are dropped (without breaking hierarchy rules). The <code>scope</code> can also be a character vector of term labels. If the argument is supplied as a formula, any <code>.</code> is interpreted relative to the formula implied by the <code>object</code> argument.
<code>scale</code>	a single numeric value containing a residual scale estimate. If missing, the scale estimate in <code>object</code> is used.

keep	a character vector of names of components that should be saved for each subset model. Only names from the set "coefficients", "fitted" and "residuals" are allowed. If keep == TRUE, the complete set is saved. The default behavior is not to keep anything.
fast	a logical value. If TRUE the robust initial estimate (used when fitting each of the reduced models) is replaced by a weighted least squares estimate using the robust weights in object.
...	additional arguments required by the generic drop1 function.

Details

This function is a method for the generic function `drop1` for class "lmRob".

Value

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

See Also

`add1`, `anova`, `drop1`, `lmRob.object`.

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
drop1(stack.rob)
```

fastcov

Orthogonalised Pariwise Covariance Estimation

Description

Compute a robust estimate of location and scatter using an orthogonalized pairwise estimator.

Usage

```
fastcov(x, control)
```

Arguments

x	a numeric matrix containing the data.
control	a list of control parameters. The utility function <code>covRob.control</code> creates a list of the control parameters and their default values. See details for the required control parameters and their default values.

Details

This function is called by the high-level function `covRob` when either the `pairwiseGK` or the `pairwiseQC` estimator is specified (via the optional arguments `estim = "pairwisegk"` or `estim = "pairwiseqc"`). It may also be of interest to power users who want to compute a pairwise estimate with a minimum of fuss.

Presently the only control parameter is the name of the pairwise estimator, either `"pairwisegk"` or `"pairwiseqc"`.

Value

a list with the following components:

<code>call</code>	an image of the call that produced the object with all the arguments named.
<code>cov</code>	a numeric matrix containing the Stahel-Donoho estimate of the covariance/correlation matrix.
<code>center</code>	a numeric vector containing the Stahel-Donoho estimate of the location vector.
<code>raw.cov</code>	a numeric matrix containing the initial robust estimate of the covariance/correlation matrix.
<code>raw.center</code>	a numeric vector containing the initial robust estimate of the location vector.

References

Alqallaf, F. A. (2003). A new contamination model for robust estimation with large high-dimensional datasets. Ph.D. Thesis. http://hajek.stat.ubc.ca/~ruben/website/Fatemah_thesis.pdf

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

See Also

`covRob`, `covRob.control`.

Examples

```
data(woodmod.dat)
X <- as.matrix(woodmod.dat)
qc.control <- covRob.control("pairwiseqc")
fastcov(X, qc.control)

gk.control <- covRob.control("pairwisegk")
fastcov(X, gk.control)
```

fastmcd

*Fast MCD Estimation***Description**

Returns a list of class `mcd` containing estimates of the robust multivariate location, the robust covariance matrix, and optionally the robust correlation matrix. Specifically, the `fastmcd` function first returns the raw minimum covariance determinant (MCD) estimator of Rousseeuw (1984, 1985). Then the MCD estimate is used to assign weights to the objects, and also weighted estimates of location and covariance are returned.

Usage

```
fastmcd(x, cor = FALSE, print.it = TRUE, quan = floor((n + p + 1)/2), ntrial = 500)
```

Arguments

<code>x</code>	a vector, matrix, or data frame. Columns represent variables, rows represent observations. Missing values (NAs) and Infinite values (Infs) are allowed. Observations (rows) with missing or infinite values are automatically excluded from the computations.
<code>cor</code>	a logical flag. If <code>cor = TRUE</code> then the estimated correlation matrix will be returned as well.
<code>print.it</code>	a logical flag. If <code>print.it = TRUE</code> information about the method will be printed.
<code>quan</code>	an integer value giving the number of observations whose covariance determinant will be minimized. The default <code>quan</code> is <code>floor((n+p+1)/2)</code> , where <code>n</code> is the number of observations and <code>p</code> is the number of variables. Any <code>quan</code> between the default and <code>n</code> may be specified.
<code>ntrial</code>	the number of random trial subsamples that are drawn for large datasets. The default is 500.

Details

Let `n` be the number of observations and `p` be the number of variables. The minimum covariance determinant estimate is given by the subset of `quan` observations of which the determinant of their covariance matrix is minimal. The MCD location estimate is then the mean of those `quan` points, and the MCD scatter estimate is their covariance matrix. The default value of `quan` is `floor((n+p+1)/2)`, but the user may choose a larger number. For multivariate data sets, it takes too much time to find the exact estimate, so an approximation is computed. A full description of the present algorithm can be found in Rousseeuw and Van Driessen (1997). Major advantages of this algorithm are its precision and the fact that it can deal with very large `n`.

Although the raw minimum covariance determinant estimate has a high breakdown value, its statistical efficiency is low. A better finite-sample efficiency can be attained while retaining the high breakdown value by computing a weighted mean and covariance estimate, with weights based on

the MCD estimate. By default, fastmcd returns both the raw MCD estimate and the weighted estimate.

Multivariate outliers can be found by means of the robust distances, as described in Rousseeuw and Leroy (1987) and in Rousseeuw and Van Zomeren (1990). These distances can be calculated by the function `mahalanobis`, and plotted by applying `plot.mcd` on a "mcd" object. It is suggested that the number of observations be at least five times the number of variables. When there are fewer observations than this, there is not enough information to determine whether outliers exist.

An important advantage of the present algorithm is that it allows for exact fit situations, where more than q observations lie on a hyperplane. Then the program still yields the MCD location and scatter matrix, the latter being singular (as it should be), as well as the equation of the hyperplane.

If the classical covariance matrix of the data is already singular, all observations lie on a hyperplane. Then fastmcd will give a message and the equation of the hyperplane. The MCD estimates are then equal to the classical estimates. In this case, you will need to modify your data before applying fastmcd, perhaps by using `princomp` and deleting columns with zero variance.

For univariate data sets, the exact algorithm `location.lts` is used. See the `location.lts` help file for more information.

Value

an object of class "mcd" with components:

<code>call</code>	an image of the call that produced the object with all the arguments named.
<code>method</code>	a character string that contains information about the method and about singular subsamples (if any).
<code>quan</code>	the number of observations that have determined the minimum covariance determinant estimator. The default is $\text{floor}((n+p+1)/2)$, where n is the number of observations and p the number of variables.
<code>mcd.wt</code>	weights based on the estimated covariance matrix and the estimated location of the data.
<code>X</code>	the input data.
<code>raw.cov</code>	the raw MCD covariance matrix.
<code>raw.center</code>	the raw MCD location of the data.
<code>raw.objective</code>	the determinant of the raw MCD covariance matrix.
<code>cov</code>	the robust covariance matrix obtained by reweighting. (If the raw MCD is singular, it is given here.)
<code>cor</code>	the estimated correlation matrix for the data. This is only returned if <code>cor = TRUE</code> .
<code>center</code>	the robust location estimate of the data, obtained by reweighting. (If the raw MCD is singular, its center is given here.)
<code>n.obs</code>	the number of data observations (after any missing values have been removed).

Side Effects

If `print.it = TRUE` a message is printed.

Background

The minimum covariance determinant estimator (Rousseeuw, 1985) has a breakdown value of roughly $(n - \text{quan}) / n$, which is about 50% for the default `quan`. That is, the estimate cannot be made arbitrarily bad without changing about half of the data. A covariance matrix is considered to be arbitrarily bad if some eigenvalue goes to infinity or to zero (singular matrix). This is analogous to a univariate scale estimate, which breaks down if the estimate is going either to infinity or to zero.

References

- Rousseeuw, P. J. and Van Driessen, K. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41, 212-223.
- Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association*, 79, 871-881.
- Rousseeuw, P. J. (1985). Multivariate estimation with high breakdown point. In *Mathematical Statistics and Applications*. W. Grossmann, G. Pflug, I. Vincze and W. Wertz, eds. Reidel: Dordrecht, 283-297.
- Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. Wiley-Interscience, New York. [Chapter 7]
- Rousseeuw, P. J. and van Zomeren, B. C. (1990). Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, 85, 633-639.

See Also

[covRob](#).

Examples

```
data(stack.dat)
fastmcd(stack.dat)
```

fit.models

Model Comparison

Description

Combines comparable models into single object. Generic methods produce side-by-side comparisons of the estimated model parameters and diagnostic plots.

Usage

```
fit.models(model.list, formula = NULL, ...)
```

Arguments

<code>model.list</code>	either a list of calls or a list of function names. This argument is not required when combining one or more fitted models into a <code>fit.models</code> object.
<code>formula</code>	when <code>model.list</code> is a list of function names the second argument is often a formula. This saves you the trouble of having to name it explicitly when calling <code>fit.models</code> .
<code>...</code>	additional arguments to be passed to the functions in <code>model.list</code> .

Details

If `model.list` is a list of calls then `fit.models` replaces each element by its evaluated call and sets the appropriate class.

If `model.list` is a list of function names then `fit.models` creates a call to each of the functions in `model.list` by combining the respective function name with the remaining arguments passed to `fit.models`. The list of calls is then evaluated as described above.

If `model.list` is a fitted model object then `fit.models` assumes that each of its argument is a fitted model object. The arguments are put into a list, the appropriate class is set, and the list is returned. To be comparable the specified models must all belong to one of the supported classes in `fit.models`. The supported classes of models in this release of the Robust Library are (1) Covariance/Correlation Models {"cov", "covRob"}. The subfunction `get.fit.models.database` provides `fit.models` with the classes of comparable models.

Value

The returned object is a list containing the fitted models. The class of the returned object depends on the classes of the model objects it contains. Returned objects containing class "cov" and "covRob" objects have class "covfm".

See Also

[get.fit.models.database](#).

Examples

```
data(woodmod.dat)
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)

#This way doesn't work in R yet
#woodmod.cls <- cov(woodmod.dat)
#woodmod.rob <- covRob(woodmod.dat)
#woodmod.fm <- fit.models(Robust = woodmod.rob, Classical = woodmod.cls)
```

```
gen.data
```

Generate Data With Contamination

Description

Generates a random dataset with some amount of contamination.

Usage

```
gen.data(coeff, n = 100, eps = 0.1, sig = 3, snr = 1/20, seed = 837)
```

Arguments

<code>coeff</code>	a numeric vector of length 3 containing the true coefficients.
<code>n</code>	a positive integer giving the number of observations in the data set.
<code>eps</code>	a numeric value between 0 and 0.5 specifying the fraction of contamination.
<code>sig</code>	a positive numeric value giving the standard deviation of the uncontaminated data.
<code>snr</code>	a positive numeric value giving the signal to noise ratio, well not really.
<code>seed</code>	an integer value giving the seed for the random number generator.

Value

a data frame with `n` rows and 4 columns. The regressors are generated as: `rnorm(n, 1)`, `rnorm(n, 1)^3`, `exp(rnorm(n, 1))`. It also generates an unused vector `x4`.

```
get.fit.models.database
```

Model Comparison Database

Description

Returns a list of comparable models.

Usage

```
get.fit.models.database()
```

Value

a list of comparable models for `fit.models`.

See Also

[fit.models](#)

Examples

```
get.fit.models.database()
```

```
glmRob.Initial.LMS Initial LMS Coefficient Estimates for CUBIF Method
```

Description

Computes initial parameter estimates for the CUBIF method using the least median of squares algorithm.

Usage

```
glmRob.Initial.LMS(X, y, ni, dist, offset, icode)
```

Arguments

X	model matrix.
y	response.
ni	ni.
dist	distances.
offset	offset.
icode	family indicator.

Value

a list containing the initial parameter and scale estimates.

See Also

[glmRob.cubif](#)

```
glmRob Fit a Robust Generalized Linear Model
```

Description

Produces an object of class glmRob which is a Robust Generalized Linear Model fit.

Usage

```
glmRob(formula, family = binomial(), data, weights, subset,  
na.action, method = "cubif", model = TRUE, x = FALSE, y = TRUE,  
control = glmRob.control, contrasts = NULL, ...)
```

Arguments

<code>formula</code>	a formula expression as for other regression models, of the form <code>response ~ predictors</code> . See the documentation of <code>lm</code> and <code>formula</code> for details.
<code>family</code>	a family object - only <code>binomial</code> and <code>poisson</code> are implemented. See the documentation of <code>glm</code> for details.
<code>data</code>	an optional data frame in which to interpret the variables occurring in the formula.
<code>weights</code>	an optional vector of weights to be used in the fitting process. Should be <code>NULL</code> or a numeric vector.
<code>subset</code>	an expression specifying the subset of the data to which the model is fit. This can be a logical vector (which is replicated to have length equal to the number of observations), a numeric vector indicating which observations are included, or a character vector of the row names to be included. By default all observations are used.
<code>na.action</code>	a function to filter missing data. This is applied to the <code>model.frame</code> after any <code>subset</code> argument has been used. The default (<code>na.fail</code>) is to create an error if any missing values are found. A possible alternative is <code>na.omit</code> which omits the rows that contain one or more missing values.
<code>method</code>	a character vector indicating the fitting method. The choices are <code>method = "cubif"</code> for the conditionally unbiased bounded influence estimator, <code>method = "mallows"</code> for Mallows' leverage downweighting estimator, and <code>method = "misclass"</code> for a consistent estimate based on the misclassification model. The Mallows' and misclassification estimators are only defined for logistic regression models with Bernoulli response.
<code>model</code>	a logical flag. If <code>TRUE</code> then the model frame is returned.
<code>x</code>	a logical flag. If <code>TRUE</code> then the model matrix is returned.
<code>y</code>	a logical flag. If <code>TRUE</code> then the response variable is returned.
<code>contrasts</code>	a list of contrasts to be used for some or all of the factors appearing as variables in the model formula. The names of the list should be the names of the corresponding variables, and the elements should either be contrast-type matrices (matrices with as many rows as levels of the factor and with columns linearly independent of each other and of a column of one's), or else they should be functions that compute such contrast matrices.
<code>control</code>	a list of iteration and algorithmic constants to control the conditionally unbiased bounded influence robust fit. See <code>glmRob.cubif.control</code> for their names and default values. These can also be set as arguments of <code>glmRob</code> itself.
<code>...</code>	control arguments maybe specified directly.

Value

a list with class `glmRob` containing the robust generalized linear model fit. See `glmRob.object` for details.

References

- Copas, J. B. (1988). Binary Regression Models for Contaminated Data. JRSS 50, 225-265.
- Kunsch, L., Stefanski L. and Carroll, R. (1989). Conditionally Unbiased Bounded-Influence Estimation in General Regression Models, with Applications to Generalized Linear Models. JASA 50, 460-466.
- Carroll, R. J. and Pederson, S. (1993). On Robustness in the Logistic Regression Model. JRSS 55, 693-706.
- Marazzi, A. (1993). Algorithms, routines and S functions for robust statistics. Wadsworth & Brooks/Cole, Pacific Grove, CA.

See Also

[glmRob.control](#), [glmRob.object](#), [glmRob.cubif.control](#), [glmRob.mallows.control](#), [glmRob.misclass.control](#), [glm](#).

Examples

```
data(breslow.dat)

glmRob(sumY ~ Age10 + Base4*Trt, family = poisson(), data = breslow.dat, method = "cubif")
```

`glmRob.control` *glmRob Control Parameters*

Description

Generates a list of control parameters for glmRob. The main purpose of this function is to implement the default behaviour for glmRob. Use the functions listed in the See Also section to generate control lists for the different robust estimators.

Usage

```
glmRob.control(method, ...)
```

Arguments

<code>method</code>	a character vector specifying which estimator the control parameters should be generated for. The choices are "cubif", "mallows", and "misclass".
<code>...</code>	additional arguments are included in the control (if appropriate for the estimator specified by <code>method</code>).

Value

a list of control parameters appropriate for the fitting method specified by the `method` argument.

See Also

[glmRob.cubif.control](#), [glmRob.mallows.control](#), [glmRob.misclass.control](#).

glmRob.cubif.Ini *CUBIF Initialization*

Description

Compute the initial coefficient estimates used in the

Usage

```
glmRob.cubif.Ini(X, y, ni, icase, offset, b, zmin, epsilon, ial, control)
```

Arguments

X	model matrix.
y	response.
ni	ni.
icase	family indicator.
offset	offset.
b	tuning parameter.
zmin	tuning parameter.
epsilon	tolerance.
ial	ial.
control	control parameters.

Value

a list containing the initial coefficient estimates.

See Also

[glmRob.cubif](#)

glmRob.cubif *CUBIF GLM Fitter*

Description

Robustly fit a generalized linear model using a conditionally unbiased bounded influence estimator. This function is called by the high-level function [glmRob](#) when `method = "cubif"` (the default) is specified.

Usage

```
glmRob.cubif(x, y, intercept = FALSE, offset = 0, family = binomial(),  
            null.dev = TRUE, control)
```

Arguments

x	a numeric model matrix.
y	either a numeric vector containing the response or, in the case of the binomial family, a two-column numeric matrix containing the number of successes and failures.
intercept	a logical value. If TRUE a column of ones is added to the design matrix.
offset	a numeric vector containing the offset.
family	a family object.
null.dev	a logical value. If TRUE the null deviance is computed.
control	a list of control parameters. See glmRob.cubif.control .

Value

See [glmRob.object](#).

References

Kunsch, L., Stefanski L. and Carroll, R. (1989). Conditionally Unbiased Bounded-Influence Estimation in General Regression Models, with Applications to Generalized Linear Models. *JASA* 84, 460-466.

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

See Also

[glmRob](#), [glmRob.cubif.control](#).

glmRob.cubif.control

Control Parameters for the Bounded Influence Robust GLM Estimator

Description

Allows users to set parameters for glmRob.

Usage

```
glmRob.cubif.control(epsilon = 0.001, maxit = 50, bpar = 2, cpar = 1.5,  
                    trc = FALSE, ...)
```

Arguments

epsilon	a positive numeric values specifying the convergence threshold for the parameters.
maxit	a positive integer giving the maximum number of iterations.
bpar	bpar
cpar	a single positive numeric value specifying the tuning constant for the initial estimate. This is the truncation value for the likelihood equation for the initial estimate. It determines the starting point of the iterative algorithm to calculate the final estimate.
trc	a logical value. If TRUE the number of the current iteration is printed on the screen.
...	additional arguments are ignored.

Value

a list is returned containing the values specified in the Arguments section.

See Also

[glmRob](#).

glmRob.cubif.null *CUBIF Null Deviance*

Description

Computes the null deviance in the CUBIF method.

Usage

```
glmRob.cubif.null(x, y, ni, offset, ics, family, control, robust.cov.control)
```

Arguments

x	model matrix.
y	response.
ni	ni.
offset	offset.
ics	ics.
family	a family object.
control	control parameters.
robust.cov.control	control parameters.

Value

a numeric value containing the null deviance.

See Also

glmRob.mallows *Mallows Type Estimator*

Description

Computes the Mallows Type Estimator provided by glmRob.

Usage

```
glmRob.mallows(x, y, control, offset, null.dev, family, Terms)
```

Arguments

x	model matrix
y	a numeric vector of Bernoulli responses.
control	control parameters.
offset	offset
null.dev	a logical value. If TRUE the null deviance is computed and stored.
family	a binomial family object.
Terms	the Terms object created in glmRob.

Value

a list similar to [glmRob.object](#).

See Also

```
link{glmRob}
```

Examples

```
data(mallows.dat)
```

```
glmRob(y ~ a + b + c, data = mallows.dat, family = binomial(), method = 'mallows')
```

```
glmRob.mallows.control
```

Control for Mallows-type Robust GLM Estimator

Description

Allows users to set parameters for `glmRob`.

Usage

```
glmRob.mallows.control(wt.fn = wt.carroll, wt.tuning = 8, ...)
```

Arguments

<code>wt.fn</code>	a weight function that might depend on a tuning constant. This function will be evaluated at the square root of the robust Mahalanobis distances of the covariates divided by their dimension.
<code>wt.tuning</code>	a tuning constant for <code>wt.fn</code> .
<code>...</code>	additional arguments are ignored.

Value

a list is returned, consisting of these parameters packaged to be used by `glmRob()`. The values for `glmRob.mallows.control()` can be supplied directly in a call to `glmRob()`. These values are filtered through `glmRob.mallows.control()` inside `glmRob()`.

See Also

[glmRob](#).

```
glmRob.misclass
```

Consistent Misclassification Estimator

Description

Computes the consistent misclassification estimate provided in [glmRob](#).

Usage

```
glmRob.misclass(x, y, control, offset, null.dev, family, Terms)
```

Arguments

x	model matrix.
y	response.
control	control parameters.
offset	offset.
null.dev	a logical value.
family	a binomial family object.
Terms	the Terms object computed in glmRob.

Value

a list similar to `glmRob.object`.

See Also

[glmRob](#)

Examples

```
data(leuk.dat)

glmRob(y ~ ag + wbc, data = leuk.dat, family = binomial(), method = 'misclass')
```

```
glmRob.misclass.control
```

Control for Misclassification Robust GLM Estimator

Description

Allows users to set parameters for `glmRob`.

Usage

```
glmRob.misclass.control(mc.gamma = 0.01, mc.maxit = 30, mc.trc = FALSE,
  mc.tol = 0.001, mc.initial = NULL, ...)
```

Arguments

mc.gamma	a real number between 0 and 1 that represents the probability of misclassification of a response variable.
mc.maxit	maximum number of iterations.
mc.trc	a logical value indicating whether a trace of the current parameter values is printed to the screen while the algorithm iterates.
mc.tol	convergence threshold.
mc.initial	a vector of initial values to start the iterations. If omitted, the coefficients resulting from a non-robust glm fit are used.
...	additional arguments are ignored.

Value

a list containing the parameters packaged to be used by `glmRob`. The values for `glmRob.misclass.control` can be supplied directly in a call to `glmRob`. These values are filtered through `glmRob.misclass.control` inside `glmRob`.

See Also

`glmRob`

`glmRob.misclass.w` *Misclass Weights*

Description

Compute the weights used by the constant misclassification estimator.

Usage

```
glmRob.misclass.w(u, lambda)
```

Arguments

<code>u</code>	a numeric vector.
<code>lambda</code>	a numeric value.

Value

a numeric vector.

See Also

`glmRob.misclass`

`glmRob.object` *Robust Generalized Linear Model Fit*

Description

These are objects of class `glmRob` which represent the robust fit of a generalized linear regression model, as estimated by `glmRob()`.

Value

<code>coefficients</code>	the coefficients of the <code>linear.predictors</code> , which multiply the columns of the model matrix. The names of the coefficients are the names of the single-degree-of-freedom effects (the columns of the model matrix). If the model is over-determined there will be missing values in the coefficients corresponding to inestimable coefficients.
<code>linear.predictors</code>	the linear fit, given by the product of the model matrix and the coefficients.
<code>fitted.values</code>	the fitted mean values, obtained by transforming <code>linear.predictors</code> using the inverse link function.
<code>residuals</code>	the residuals from the final fit; also known as working residuals, they are typically not interpretable.
<code>deviance</code>	up to a constant, minus twice the log-likelihood evaluated at the final <code>coefficients</code> . Similar to the residual sum of squares.
<code>null.deviance</code>	the deviance corresponding to the model with no predictors.
<code>family</code>	a 3 element character vector giving the name of the family, the link and the variance function.
<code>rank</code>	the number of linearly independent columns in the model matrix.
<code>df.residuals</code>	the number of degrees of freedom of the residuals.
<code>call</code>	a copy of the call that produced the object.
<code>assign</code>	the same as the <code>assign</code> component of an "lm" object.
<code>contrasts</code>	the same as the <code>contrasts</code> component of an "lm" object.
<code>terms</code>	the same as the <code>terms</code> component of an "lm" object.
<code>ni</code>	vector of the number of repetitions on the dependent variable. If the model is poisson then <code>ni</code> is a vector of 1s.
<code>weights</code>	weights from the final fit.
<code>iter</code>	number of iterations used to compute the estimates.
<code>y</code>	the dependent variable.
<code>contrasts</code>	the same as the <code>contrasts</code> term of an "lm" object. The object will also contain other components related to the numerical fit that are not relevant for the associated methods.

Methods

`anova`, `coefficients`, `deviance`, `fitted.values`, `family`, `formula`, `plot`, `print`, `residuals`, `summary`.

Structure

The following components must be included in a legitimate "glmRob" object. Residuals, fitted values, and coefficients should be extracted by the generic functions of the same name, rather than by the "\$" operator. The `family` function returns the entire family object used in the fitting, and `deviance` can be used to extract the deviance of the fit.

See Also

[glmRob.](#)

key

Plot Key

Description

Don't use this function. It's only here because the lattice package doesn't provide a key function. Use legend instead.

Usage

```
key(x, y, text, lines, corner = c(0, 1), cex = par("cex"), col = par("col"), lty
```

Arguments

x	x
y	y
text	text
lines	lines
corner	corner
cex	cex
col	col
lty	lty
lwd	lwd
pch	pch
...	dots

Side Effects

Adds key to plot.

`leuk.dat`*Leuk Data*

Description

An example data set for the misclassification fitter in `glmRob`.

Usage

```
data(leuk.dat)
```

Format

A data frame with 33 observations on the following 3 variables.

wbc a numeric vector.

ag a numeric vector.

y a numeric vector.

Source

Don't know - if you know please email the package maintainer.

Examples

```
data(leuk.dat)
```

`lmRob.RFPE`*Robust Final Prediction Errors*

Description

Computes the robust Final Prediction Errors (FPE) for a robust regression fit using M-estimates.

Usage

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

Arguments

`object` an `lmRob` object.

`scale` a numeric value specifying the scale estimate used to compute the robust FPE. Usually this should be the scale estimate from an encompassing model. If `NULL`, the scale estimate in `object` is used.

Value

a single numeric value giving the robust final prediction error.

See Also

`lmRob`, `step.lmRob`, `add1.lmRob`, `drop1.lmRob`.

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
lmRob.RFPE(stack.rob)
```

lmRob

High Breakdown and High Efficiency Robust Linear Regression

Description

Performs a robust linear regression with high breakdown point and high efficiency regression.

Usage

```
lmRob(formula, data, weights, subset, na.action, model = TRUE, x = FALSE, y = FALSE)
```

Arguments

<code>formula</code>	a formula object, with the response on the left of a <code>~</code> operator, and the terms, separated by <code>+</code> operators, on the right.
<code>data</code>	a <code>data.frame</code> in which to interpret the variables named in the <code>formula</code> , or in the <code>subset</code> and the <code>weights</code> argument. If this is missing, then the variables in the <code>formula</code> should be on the search list. This may also be a single number to handle some special cases - see below for details.
<code>weights</code>	vector of observation weights; if supplied, the algorithm fits to minimize the sum of a function of the square root of the weights multiplied into the residuals. The length of <code>weights</code> must be the same as the number of observations. The weights must be nonnegative and it is strongly recommended that they be strictly positive, since zero weights are ambiguous, compared to use of the <code>subset</code> argument.
<code>subset</code>	expression saying which subset of the rows of the data should be used in the fit. This can be a logical vector (which is replicated to have length equal to the number of observations), or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
<code>na.action</code>	a function to filter missing data. This is applied to the <code>model.frame</code> after any <code>subset</code> argument has been used. The default (with <code>na.fail</code>) is to create an error if any missing values are found. A possible alternative is <code>na.exclude</code> , which deletes observations that contain one or more missing values.

<code>model</code>	a logical flag: if TRUE, the model frame is returned in component <code>model</code> .
<code>x</code>	a logical flag: if TRUE, the model matrix is returned in component <code>x</code> .
<code>y</code>	a logical flag: if TRUE, the response is returned in component <code>y</code> .
<code>contrasts</code>	a list giving contrasts for some or all of the factors appearing in the model formula. The elements of the list should have the same name as the variable and should be either a contrast matrix (specifically, any full-rank matrix with as many rows as there are levels in the factor), or else a function to compute such a matrix given the number of levels.
<code>nrep</code>	the number of random subsamples to be drawn. If "Exhaustive" resampling is being used, the value of <code>nrep</code> is ignored.
<code>control</code>	a list of control parameters to be used in the numerical algorithms. See lmRob.control for the possible control parameters and their default settings.
<code>genetic.control</code>	a list of control parameters to be used in the genetic algorithm, if chosen.
<code>...</code>	additional arguments are passed to the <code>ccontrol</code> functions.

Details

By default, the `lmRob` function automatically chooses an appropriate algorithm to compute a final robust estimate with high breakdown point and high efficiency. The final robust estimate is computed based on an initial estimate with high breakdown point. For the initial estimation, the alternate M-S estimate is used if there are any factor variables in the predictor matrix, and an S-estimate is used otherwise. To compute the S-estimate, a random resampling or a fast procedure is used unless the data set is small, in which case exhaustive resampling is employed. See [lmRob.control](#) for how to choose between the different algorithms.

Value

a list describing the regression. Note that the solution returned here is an approximation to the true solution based upon a random algorithm (except when "Exhaustive" resampling is chosen). Hence you will get (slightly) different answers each time if you make the same call with a different seed. See [lmRob.control](#) for how to set the seed, and see [lmRob.object](#) for a complete description of the object returned.

References

- Gervini, D., and Yohai, V. J. (1999). A class of robust and fully efficient regression estimates, mimeo, Universidad de Buenos Aires.
- Marazzi, A. (1993). *Algorithms, routines, and S functions for robust statistics*. Wadsworth & Brooks/Cole, Pacific Grove, CA.
- Maronna, R. A., and Yohai, V. J. (1999). Robust regression with both continuous and categorical predictors, mimeo, Universidad de Buenos Aires.
- Pena, D., and Yohai, V. (1999). A Fast Procedure for Outlier Diagnostics in Large Regression Problems, *Journal of the American Statistical Association*, 94, 434-445.
- Yohai, V. (1988). High breakdown-point and high efficiency estimates for regression, *Annals of Statistics*, 15, 642-665.

Yohai, V., Stahel, W. A., and Zamar, R. H. (1991). A procedure for robust estimation and inference in linear regression, in Stahel, W. A. and Weisberg, S. W., Eds., *Directions in robust statistics and diagnostics, Part II*. Springer-Verlag.

See Also

[lmRob.control](#), [lmRob.object](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
```

<code>lmRob.control</code>	<i>Control Parameters for Robust Linear Regression</i>
----------------------------	--

Description

Allows the users to set values affecting the estimation procedure for robust regression in `lmRob`.

Usage

```
lmRob.control(tlo = 1e-4, tua = 1.5e-06, mxr = 50, mxf = 50, mxs = 50, tl = 1e-06,
```

Arguments

<code>tlo</code>	the relative tolerance in the iterative algorithms.
<code>tua</code>	the tolerance used for the determination of pseudo-rank.
<code>mxr</code>	the maximum number of iterations in the refinement step.
<code>mxf</code>	the maximum number of iterations for computing final coefficient estimates.
<code>mxs</code>	the maximum number of iterations for computing scale estimate.
<code>tl</code>	the tolerance for scale denominators. If a scale estimate becomes less than <code>tl</code> , the scale estimate is set equal to <code>tl</code> .
<code>estim</code>	parameter that determines the type of estimator to be computed. If <code>estim="Initial"</code> , only the initial estimates are computed; if <code>estim="Final"</code> , then final estimates are returned.
<code>initial.alg</code>	parameter that determines the algorithm for initial estimates. Valid choices are "Auto" for data-dependent algorithm, "Random" for random resampling, "Exhaustive" for exhaustive resampling, "Fast" for fast procedure, and "Genetic" for genetic algorithm. By default, <code>lmRob</code> uses "Auto".
<code>final.alg</code>	parameter that determines the type of the final estimates. Valid choices are "Adaptive" for the robust efficient weighted least squares as proposed in Gervini and Yohai (1999), and "MM" for MM-estimate as proposed in Yohai, Stahel and Zamar (1991). By default, <code>lmRob</code> uses "MM".

seed	seed parameter used in the random sampling and genetic algorithm for the computation of initial estimates.
weight	a character vector that determines the type of loss functions to be used. The first determines the loss function used for the initial estimates, and the second determines the loss function used for the final M-estimates. Valid choices are "Optimal" and "Bisquare".
level	the level of significance of the test for bias of the final MM-estimates, if desired later on.
efficiency	the asymptotic efficiency of the final estimate.
trace	a logical flag: if TRUE, the remaining computing time will be printed.

Value

a list containing the values used for each of the control parameters.

See Also

[lmRob](#).

Examples

```
data(stack.dat)
my.control <- lmRob.control(weight=c("Bisquare", "Optimal"))
stack.bo <- lmRob(Loss ~ ., data = stack.dat, control = my.control)
```

lmRob.effvy

Constant for the Optimal Loss (Weight) Function

Description

Returns the constant of the optimal loss (weight) function, given the asymptotic efficiency under Gaussian model.

Usage

```
lmRob.effvy(eff, ipsi = 1)
```

Arguments

eff	a numeric value between 0 and 1 specifying the desired asymptotic efficiency.
ipsi	choose your weight function: ipsi = 1 for optimal function, ipsi = 2 for rescaled bisquare function, and ipsi = 3 for Huber function.

Value

the constant of the optimal loss (weight) function which will give the required asymptotic efficiency.

See Also

[lmRob](#).

lmRob.fit.compute *Fit a Robust Linear Model*

Description

Fits a robust linear model with high breakdown point and high efficiency estimates. This is used by [lmRob](#), but not supposed to be called by the users directly.

Usage

```
lmRob.fit.compute(x2, y, x1 = NULL, x1.idx = NULL, nrep = NULL, robust.control = NU
```

Arguments

x2	numeric vector or matrix for the continuous predictors.
y	numeric vector for the response in a linear model.
x1	numeric vector or matrix for the discrete predictors.
x1.idx	numeric vector giving the index of x1 as column numbers of the whole predictor matrix.
nrep	the number of random subsamples to be drawn. If "Exhaustive" resampling is being used, the value of nrep is ignored.
robust.control	a list of control parameters to be used in the numerical algorithms. See <code>lmRob.control</code> for the possible control parameters and their default settings.
genetic.control	a list of control parameters to be used in the genetic algorithm, if chosen.
...	additional arguments.

Value

an object of class "lmRob". See [lmRob.object](#) for a complete description of the object returned.

References

- Gervini, D., and Yohai, V. J. (1999). A class of robust and fully efficient regression estimates, mimeo, Universidad de Buenos Aires.
- Marazzi, A. (1993). *Algorithms, routines, and S functions for robust statistics*. Wadsworth & Brooks/Cole, Pacific Grove, CA.
- Maronna, R. A., and Yohai, V. J. (1999). Robust regression with both continuous and categorical predictors, mimeo, Universidad de Buenos Aires.

Yohai, V. (1988). High breakdown-point and high efficiency estimates for regression, *Annals of Statistics*, 15, 642-665.

Yohai, V., Stahel, W. A., and Zamar, R. H. (1991). A procedure for robust estimation and inference in linear regression, in Stahel, W. A. and Weisberg, S. W., Eds., *Directions in robust statistics and diagnostics, Part II*. Springer-Verlag.

See Also

[lmRob](#), [lmRob.control](#).

lmRob.ga

Estimation of S-Estimates with Genetic Algorithm

Description

Computes the robust S-estimates using a genetic algorithm. It is used by `lmRob`, not supposed to be called by the users directly.

lmRob.object

Robust Linear Model Objects

Description

These are objects of class `lmRob` which represent the robust fit of a linear regression model, as estimated by `lmRob` function.

Value

`coefficients` vector of coefficients for the robust regression. If `est="final"`, these are final estimates; if `est="initial"`, these are initial estimates.

`T.coefficients` the vector of coefficients for the initial estimate, if `est="final"`.

`scale` the scale estimate computed using the initial estimates.

`residuals` the residual vector corresponding to the estimates returned in `coefficients`.

`T.residuals` the residual vector corresponding to the estimates returned in `T.coefficients`.

`fitted.values` the fitted values corresponding to the estimates returned in `coefficients`.

`T.fitted.values` the fitted values corresponding to the estimates returned in `T.coefficients`.

`cov` the estimated covariance matrix of the estimates in `coefficients`.

`T.cov` the estimated covariance matrix of the estimates in `T.coefficients`.

`rank` the rank of the design matrix x .

<code>iter.refinement</code>	the number of iterations required to refine the initial estimates.
<code>df.residuals</code>	the degrees of freedom in the residuals (the number of rows in <code>x</code> minus the rank of <code>x</code>).
<code>est</code>	a character string that specifies the type of estimates returned. If <code>est="initial"</code> , the initial estimates are returned; if <code>est="final"</code> , the final estimates are returned.
<code>control</code>	a list of control parameters, passed to the function <code>lmRob</code> as the <code>robust.control</code> argument that produced the <code>lmRob</code> object.
<code>genetic.control</code>	a list of control parameters, passed to the function <code>lmRob</code> as the <code>genetic.control</code> argument that produced the <code>lmRob</code> object, if present.
<code>dev</code>	the robust deviance if final MM-estimates are returned.
<code>T.dev</code>	the robust deviance corresponding to initial S-estimates if applies.
<code>r.squared</code>	the fraction of variation in <code>y</code> explained by the robust regression on <code>x</code> corresponding to the final MM-estimates in <code>coefficients</code> , if applies.
<code>T.r.squared</code>	the fraction of variation in <code>y</code> explained by the robust regression on <code>x</code> corresponding to the initial S-estimates in <code>T.coefficients</code> , if applies.
<code>M.weights</code>	the robust estimate weights corresponding to the final MM-estimates in <code>coefficients</code> , if applies.
<code>T.M.weights</code>	the robust estimate weights corresponding to the initial S-estimates in <code>T.coefficients</code> , if applies.
<code>iter.final.coef</code>	the number of iterations required to compute the final MM-estimates of the coefficients, if applies.
<code>call</code>	an image of the call that produced the object, but with the arguments all named and with the actual formula included as the <code>formula</code> argument.
<code>assign</code>	the same as the <code>assign</code> component of an "lm" object.
<code>contrasts</code>	the same as the <code>contrasts</code> component of an "lm" object.
<code>terms</code>	the same as the <code>terms</code> component of an "lm" object.

Generation

This class of objects is returned from the `lmRob` function.

Methods

[add1](#), [anova](#), [coef](#), [deviance](#), [dropl](#), [fitted](#), [formula](#), [labels](#), [plot](#), [print](#), [residuals](#), [summary](#), [update](#).

Structure

The following components must be included in a legitimate "lmRob" object:

See Also

[lmRob](#).

lmRob.ucovcoef	<i>Unscaled Covariance Matrix of Coefficient Estimates</i>
----------------	--

Description

Computes the unscaled covariance matrix of the coefficient estimates. It is used by `lmRob`, not supposed to be called by the users directly.

lmfm2DRegPlot	<i>Data with Fit</i>
---------------	----------------------

Description

Plots the linear model fits in an `lmfm` object over a scatterplot of the data.

Usage

```
lmfm2DRegPlot(x, cutoff = TRUE, main, xlab, ylab, ...)
```

Arguments

<code>x</code>	an <code>lmfm</code> object.
<code>cutoff</code>	a logical value. If <code>TRUE</code> then the threshold beyond which points receive zero weight is plotted with dashed lines.
<code>main</code>	a character string specifying the plot title.
<code>xlab</code>	a character string specifying the x-axis label.
<code>ylab</code>	a character string specifying the y-axis label.
<code>...</code>	additional arguments are passed to the low-level plotting functions.

Value

`x` is invisibly returned.

lmfmOverlaidQQPlot *Overlaid QQ Plot*

Description

Overlaid residual qqplots for each of the models in an lmfm object.

Usage

```
lmfmOverlaidQQPlot(x, main, xlab, ylab, ...)
```

Arguments

x	an lmfm object.
main	a character string specifying the plot title.
xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

x is invisibly returned.

lmfmOverlaidResDenPlot
Overlaid Residual Density Plot

Description

Overlaid residual density plots for each of the models in an lmfm object.

Usage

```
lmfmOverlaidResDenPlot(x, main, xlab, ylab, ...)
```

Arguments

x	an lmfm object.
main	a character string specifying the plot title.
xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

x is invisibly returned.

 lmfmResKernDenPlot *Residual Kernel Density Plot*

Description

Produces side-by-side residual kernel density plots from an lmfm object.

Usage

```
lmfmResKernDenPlot(x, ...)
```

Arguments

`x` an lmfm object.
`...` additional arguments are passed to the low-level plotting functions.

Value

`x` is invisibly returned.

 lmfmResQQPlot *Residual QQ-Plots with Confidence Envelope*

Description

Produces side-by-side residual qq-plots from an lmfm object. A simulated confidence envelope is provided for each plot.

Usage

```
lmfmResQQPlot(x, type = "response", envelope = TRUE, half.normal = FALSE, n.samples
```

Arguments

`x` an lmfm object.
`type` a character string specifying the type of residuals used in the plot. This argument is passed to the generic residuals function when extracting the residuals from `x`.
`envelope` a logical value. If TRUE a level confidence envelope is simulated for each qq-plot.
`half.normal` a logical value. If TRUE the plot is drawn using the absolute values of the residuals.
`n.samples` a positive integer value giving the number of samples to compute in the simulation of the confidence envelope.

level	a numeric value between 0 and 1 specifying the confidence level for the envelope.
id.n	a non-negative integer value specifying the number of extreme points to identify.
robustQQline	a logical value. If TRUE then a qq-line computed using <code>lmRob</code> is included in the plot.
main	a character string specifying the plot title.
xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

`x` is invisibly returned.

```
lmfmResVsFittedPlot
```

Residuals versus Fitted Values Plot

Description

Produces side-by-side residuals versus fitted values plots from an `lmfm` object.

Usage

```
lmfmResVsFittedPlot(x, type = "response", smooths = FALSE, rugplot = FALSE, id.n =
```

Arguments

<code>x</code>	an <code>lmfm</code> object.
<code>type</code>	a character string specifying the type of residuals used in the plot. This argument is passed to the generic residuals function when extracting the residuals from <code>x</code> .
<code>smooths</code>	a logical value. If TRUE then a smooth curve computed using <code>loess</code> is included in each panel of the plot.
<code>rugplot</code>	a logical value. If TRUE then a rugplot is included in each panel of the plot.
<code>id.n</code>	a non-negative integer value giving the number of extreme points to identify.
<code>main</code>	a character string specifying the plot title.
<code>xlab</code>	a character string specifying the x-axis label.
<code>ylab</code>	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

`x` is invisibly returned.

 lmfmRespVsFittedPlot

Response versus Fitted Values Plot

Description

Produces side-by-side response versus fitted value plots from an lmfm object.

Usage

```
lmfmRespVsFittedPlot(x, smooths = FALSE, rugplot = FALSE, ...)
```

Arguments

<code>x</code>	an lmfm object.
<code>smooths</code>	a logical value. If TRUE then a smooth curve computed by loess is added to each panel of the plot.
<code>rugplot</code>	a logical value. If TRUE then a rugplot is added to each panel of the plot.
<code>...</code>	additional arguments are passed to the low-level plotting functions.

Value

`x` is invisibly returned.

 lmfmSRvsRDPlot

Standardized Residuals versus Robust Distances Plot

Description

Produces side-by-side residuals versus robust distances plots from an lmfm object.

Usage

```
lmfmSRvsRDPlot(x, type = "response", level = 0.95, id.n = 3, main, xlab, ylab, ...)
```

Arguments

<code>x</code>	an lmfm object.
<code>type</code>	a character string specifying the type of residuals used in the plot. This argument is passed to the generic residuals function when extracting the residuals from <code>x</code> .
<code>level</code>	a numeric value between 0 and 1 giving the confidence level used to draw the threshold for both the distances and residuals axes.
<code>id.n</code>	a non-negative integer value specifying the number of extreme points to identify.
<code>main</code>	a character string specifying the plot title.

xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

x is invisibly returned.

```
lmfmSqrtResVsFittedPlot
```

Squared Root of Residuals versus Fitted Values Plot

Description

Produces side-by-side square root of residuals versus fitted values plots from an lmfm object.

Usage

```
lmfmSqrtResVsFittedPlot(x, type = "response", smooths = FALSE, rugplot = FALSE, id.
```

Arguments

x	an lmfm object.
type	a character string specifying the type of residuals used in the plot. This argument is passed to the generic residuals function when extracting the residuals from x.
smooths	a logical value. If TRUE then a smooth curve computed using <code>loess</code> is included in each panel of the plot.
rugplot	a logical value. If TRUE then a rugplot is included in each panel of the plot.
id.n	a non-negative integer value giving the number of extreme points to identify.
main	a character string specifying the plot title.
xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

x is invisibly returned.

lmfmStdResPlot *Standardized Residual Plot*

Description

Produces side-by-side standardized residual plots from an lmfm object.

Usage

```
lmfmStdResPlot(x, type = "response", level = 0.95, id.n = 3, main, xlab, ylab, ...)
```

Arguments

x	an lmfm object.
type	a character string specifying the type of residuals used in the plot. This argument is passed to the generic residuals function when extracting the residuals from x.
level	a numeric value between 0 and 1 specifying the confidence level used to draw the threshold in the plot.
id.n	a non-negative integer value specifying the number of extreme points to identify.
main	a character string specifying the plot title.
xlab	a character string specifying the x-axis label.
ylab	a character string specifying the y-axis label.
...	additional arguments are passed to the low-level plotting functions.

Value

x is invisibly returned.

mallows.dat *Mallows Data*

Description

An example data set for the mallows fitter in glmRob.

Usage

```
data(mallows.dat)
```

Format

A data frame with 70 observations on the following 4 variables.

y a numeric vector.

a a numeric vector.

b a numeric vector.

c a numeric vector.

Source

Don't know - if you know please email the package maintainer.

Examples

```
data(mallows.dat)
```

```
methods.glmRob
```

Methods for glmRob Generic Functions

Description

Accessor methods for the coefficients, deviance, formula, labels, model frame, model matrix, residuals, and weights in an [lmRob](#) object.

Usage

```
## S3 method for class 'glmRob':
coef(object, ...)
## S3 method for class 'glmRob':
family(object, ...)
## S3 method for class 'glmRob':
labels(object, ...)
## S3 method for class 'glmRob':
model.frame(formula, ...)
## S3 method for class 'glmRob':
model.matrix(object, ...)
## S3 method for class 'glmRob':
residuals(object, type = c("deviance", "pearson", "working", "response"), ...)
```

Arguments

<code>formula</code>	an <code>lmRob</code> object.
<code>object</code>	an <code>lmRob</code> object.
<code>type</code>	a character string specifying the type of residuals which should be returned.
<code>...</code>	additional arguments required by the generic functions.

Value

A vector, matrix or formula containing the requested component of the glmRob object.

See Also

[lmRob](#), [lmRob.object](#).

Examples

```
data(breslow.dat)
bres.rob <- glmRob(sumY ~ Age10 + Base4 * Trt, family = poisson(), data = breslow.dat)
coef(bres.rob)
family(bres.rob)
labels(bres.rob)
model.frame(bres.rob)
model.matrix(bres.rob)
residuals(bres.rob)
```

methods.lmRob

Methods for lmRob Generic Functions

Description

Accessor methods for the coefficients, deviance, formula, labels, model frame, model matrix, residuals, and weights in an [lmRob](#) object.

Usage

```
## S3 method for class 'lmRob':
coef(object, ...)
## S3 method for class 'lmRob':
deviance(object, ...)
## S3 method for class 'lmRob':
formula(x, ...)
## S3 method for class 'lmRob':
labels(object, ...)
## S3 method for class 'lmRob':
model.frame(formula, ...)
## S3 method for class 'lmRob':
model.matrix(object, ...)
## S3 method for class 'lmRob':
residuals(object, ...)
## S3 method for class 'lmRob':
weights(object, ...)
```

Arguments

formula	an lmRob object.
object	an lmRob object.
x	an lmRob object.
...	additional arguments required by the generic functions.

Value

A vector, matrix or formula containing the requested component of the lmRob object.

See Also

[lmRob](#), [lmRob.object](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
coef(stack.rob)
deviance(stack.rob)
formula(stack.rob)
labels(stack.rob)
model.frame(stack.rob)
model.matrix(stack.rob)
residuals(stack.rob)
weights(stack.rob)
```

panel.addons

Add Extras to fit.models Plots

Description

This is a utility function used by some of the plotting functions in the Robust Library. It optionally adds a smooth fit using loess, a rugplot, and/or identifies a specified number of extreme observations.

Usage

```
panel.addons(x, y, smooths = FALSE, rugplot = FALSE, id.n = 3, ...)
```

Arguments

x	a numeric vector of x-coordinates.
y	a numeric vector of y-coordinates. Must be the same length as x.
smooths	a logical value. If TRUE then a smooth fit computed by loess is included in each panel.

`rugplot` a logical value. If TRUE then a rugplot is included in each panel.
`id.n` a non-negative integer value specifying the number of extreme points to label in each panel.
`...` additional arguments are passed to the low-level plotting functions.

Value

a NULL value is invisibly returned.

<code>plot.covfm</code>	<i>Plot Method</i>
-------------------------	--------------------

Description

The generic plot method for objects of class "cov", "covRob", and "covfm".

Usage

```
## S3 method for class 'cov':
plot(x, which.plots = "ask", ...)
## S3 method for class 'covRob':
plot(x, which.plots = "ask", ...)
## S3 method for class 'covfm':
plot(x, which.plots = "ask", ...)
```

Arguments

`x` an object of class "cov", "covRob", or "covfm".
`which.plots` either "ask", "all", or an integer vector specifying which plots to draw. If `which.plots` is an integer vector, use the plot numbers given here (or in the "ask" menu). The plot options are (2) Eigenvalues of Covariance Estimate, (3) Sqrt of Mahalanobis Distances, (4) Ellipses Matrix, and (5) Distance - Distance Plot.
`...` additional arguments to be passed to the plotting subfunctions.

Details

The actual plot functions are only implemented for "fit.models" objects. When this method is dispatched on an object of class "cov" or "covRob" the object is cast as a "fit.models" object containing a single element and plotted with `plot.covfm`. The actual plotting is done by the subfunctions listed in the See Also section.

Value

`x` is invisibly returned.

Side Effects

The requested plots are drawn on a graphics device.

See Also

`plot`, `cov`, `covRob`, `fit.models`, `covfmDistance2Plot`, `covfmEllipsesPlot`, `covfmScreePlot`, `covfmSqrtMDPlot`.

Examples

```
data(woodmod.dat)
woodmod.cov <- cov(woodmod.dat)
plot(woodmod.cov, which.plots = 1)

woodmod.covRob <- covRob(woodmod.dat)
plot(woodmod.covRob, which = 1)

woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
plot(woodmod.fm, which = 1)
```

plot.glmRob

Diagnostic Regression Plots

Description

Creates a set plots useful for assessing a robustly fitted generalized linear model. The plot options are (2) Deviance Residuals vs. Predicted Values, (3) Response vs. Predicted Values, (4) Normal QQ Plot of Pearson Residuals, (5) QQ Plot of Deviance Residuals, (6) Standardized Deviance Residuals vs. Robust Distances, (7) Standardized Deviance Residuals vs. Index (Time), and (8) Sqrt of abs(Deviance Residuals) vs. Fitted Values.

Usage

```
## S3 method for class 'glmRob':
plot(x, which.plots = "ask", ...)
```

Arguments

<code>x</code>	a <code>glmRob</code> object.
<code>which.plots</code>	either "ask", "all", or an integer vector specifying which plots to draw. If <code>which.plots</code> is an integer vector, use the plot numbers given in the description above (or in the "ask" menu).
<code>...</code>	additional arguments are pass to the plotting subfunctions which are listed in the see also section.

Details

This function casts the `glmRob` object as an `glmfm` object containing a single model. The actual plotting is then done by the function `plot.glmfm`.

Value

`x` is invisibly returned.

Side Effects

The selected plots are drawn on a graphics device.

References

Atkinson, A. C. (1985). *Plots, Transformations and Regression*. New York: Oxford University Press.

See Also

`plot`, `glmRob`, `plot.glmfm`.

`plot.glmfm`

Comparison Plots for Generalized Linear Models

Description

Create a set plots useful for comparing the fitted generalized linear models in a `glmfm` object. The plot options are (2) Deviance Residuals vs. Fitted Values, (3) Response vs. Fitted Values, (4) Normal QQ Plot of Pearson Residuals, (5) QQ Plot of Deviance Residuals, (6) Standardized Deviance Residuals vs. Robust Distances, (7) Standardized Deviance Residuals vs. Index (Time), (8) Sqrt of abs(Deviance Residuals) vs. Fitted Values.

Usage

```
plot.glmfm(x, which.plots = "ask", ...)
```

Arguments

<code>x</code>	a <code>glmfm</code> object.
<code>which.plots</code>	either "ask", "all", or an integer vector specifying which plots to draw. If <code>which.plots</code> is an integer vector, use the plot numbers given in the description above (or in the "ask" menu).
<code>...</code>	additional arguments are passed to the plotting subfunctions which are listed in the see also section.

Details

This function is a wrapper for the plot functions listed in the see also section. For finer control you may wish to call these functions directly.

Value

`x` is invisibly returned.

Side Effects

The selected plots are drawn on a graphics device.

References

Atkinson, A. C. (1985). Plots, Transformations and Regression. New York: Oxford University Press.

See Also

[plot](#), [fit.models](#), [lmfmResVsFittedPlot](#), [lmfmRespVsFittedPlot](#), [lmfmResQQPlot](#), [lmfmSRvsRDPlot](#), [lmfmStdResPlot](#), [lmfmSqrtResVsFittedPlot](#).

plot.lmRob

Diagnostic Regression Plots

Description

Creates a set plots useful for assessing a robustly fitted linear model. The plot options are (2) Normal QQ-Plot of Residuals, (3) Estimated Kernel Density of Residuals, (4) Robust Residuals vs Robust Distances, (5) Residuals vs Fitted Values, (6) Sqrt of abs(Residuals) vs Fitted Values, (7) Response vs Fitted Values, (8) Standardized Residuals vs Index (Time), (9) Overlaid Normal QQ-Plot of Residuals, and (10) Overlaid Estimated Density of Residuals. For simple linear regression models there is also the option to have a side-by-side plots of the the fit over a scatter plot of the data.

Usage

```
## S3 method for class 'lmRob':
plot(x, which.plots = "ask", ...)
```

Arguments

<code>x</code>	an <code>lmRob</code> object.
<code>which.plots</code>	either "ask", "all", or an integer vector specifying which plots to draw. If <code>which.plots</code> is an integer vector, use the plot numbers given in the description above (or in the "ask" menu).
<code>...</code>	additional arguments are pass to the plotting subfunctions which are listed in the see also section.

Details

This function casts the `lmRob` object as an `lmfm` object containing a single model. The actual plotting is then done by the function `plot.lmfm`.

Value

`x` is invisibly returned.

Side Effects

The selected plots are drawn on a graphics device.

References

Atkinson, A. C. (1985). *Plots, Transformations and Regression*. New York: Oxford University Press.

See Also

`plot`, `lmRob`, `plot.lmfm`.

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
plot(stack.rob, which.plots = 6)
```

`plot.lmfm`*Comparison Plots for Linear Regression Models*

Description

Creates a set plots useful for comparing the fitted linear models in an `lmfm` object. The plot options are (2) Normal QQ-Plot of Residuals, (3) Estimated Kernel Density of Residuals, (4) Robust Residuals vs Robust Distances, (5) Residuals vs Fitted Values, (6) Sqrt of `abs(Residuals)` vs Fitted Values, (7) Response vs Fitted Values, (8) Standardized Residuals vs Index (Time), (9) Overlaid Normal QQ-Plot of Residuals, and (10) Overlaid Estimated Density of Residuals. For simple linear regression models there is also the option to have a side-by-side plots of the the fit over a scatter plot of the data.

Usage

```
## S3 method for class 'lmfm':
plot(x, which.plots = "ask", ...)
```

Arguments

`x` an `lmfm` object.

`which.plots` either "ask", "all", or an integer vector specifying which plots to draw. If `which.plots` is an integer vector, use the plot numbers given in the description above (or in the "ask" menu).

... additional arguments are passed to the plotting subfunctions which are listed in the see also section.

Details

This function is a wrapper for the `lmfm*Plot` functions listed in the see also section. For finer control you may wish to call these functions directly.

Value

`x` is invisibly returned.

Side Effects

The selected plots are drawn on a graphics device.

References

Atkinson, A. C. (1985). *Plots, Transformations and Regression*. New York: Oxford University Press.

See Also

[plot](#), [fit.models](#), [lmfm2DRegPlot](#), [lmfmResQQPlot](#), [lmfmSqrtResVsFittedPlot](#), [lmfmOverlaidQQPlot](#), [lmfmResVsFittedPlot](#), [lmfmStdResPlot](#), [lmfmOverlaidResDenPlot](#), [lmfmRespVsFittedPlot](#), [lmfmResKernDenPlot](#), [lmfmSRvsRDPlot](#).

Examples

```
data(stack.dat)
stack.fm <- fit.models(list(Robust = "lmRob", LS = "lm"), Loss ~ ., data = stack.dat)
plot(stack.fm, which.plots = 2)
```

predict.glmRob

Predict Method for Robust Generalized Linear Model Fits

Description

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

Usage

```
predict.glmRob(object, newdata, type = c("link", "response", "terms"), se.fit = FALSE)
```

Arguments

<code>object</code>	a <code>glmRob</code> object.
<code>newdata</code>	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
<code>type</code>	a character string specifying the type of prediction. The choices are "link" for predictions on the scale of the linear predictor, "response" for predictions on the scale of the response, and "terms" which returns a matrix giving the fitted values for each term in the model formula on the scale of the linear predictor.
<code>se.fit</code>	a logical value. If <code>TRUE</code> then standard errors for the predictions are computed.
<code>terms</code>	when <code>type = "terms"</code> all terms are returned. A character vector specifies which terms are to be returned.
<code>dispersion</code>	the dispersion of the generalized linear model fit to be assumed in computing the standard errors. If omitted, that returned by 'summary' applied to the object is used.
<code>...</code>	additional arguments required by the generic predict method.

Value

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

<code>fit</code>	Predictions
<code>se.fit</code>	Estimated standard errors

See Also

[glmRob](#), [predict](#).

Examples

```
data(breslow.dat)
bres.rob <- glmRob(sumY ~ Age10 + Base4 * Trt, family = poisson(), data = breslow.dat)
predict(bres.rob)
```

predict.lmRob *Use predict() on an lmRob Object*

Description

Extracts the fitted values from an `lmRob` object and returns a matrix of predictions.

Usage

```
## S3 method for class 'lmRob':
predict(object, newdata, type = "response", se.fit = FALSE, terms = labels(object),
```

Arguments

<code>object</code>	an <code>lmRob</code> object.
<code>newdata</code>	a data frame containing the values at which predictions are required. This argument can be missing, in which case predictions are made at the same values used to compute the object. Only those predictors referred to in the right side of the formula in <code>object</code> need be present by name in <code>newdata</code> .
<code>type</code>	a single character value specifying the type of prediction. The only choice is "response". If "response" is selected, the predictions are on the scale of the response.
<code>se.fit</code>	a logical value. If <code>TRUE</code> , pointwise standard errors are computed along with the predictions.
<code>terms</code>	this argument is presently unused.
<code>...</code>	additional arguments required by the generic <code>predict</code> function.

Value

a vector of predictions, or a list consisting of the predictions and their standard errors if `se.fit = TRUE`.

Warning

`predict` can produce incorrect predictions when the `newdata` argument is used if the formula in `object` involves *data-dependent* transformations, such as `poly(Age, 3)` or `sqrt(Age - min(Age))`.

See Also

[lmRob](#), [predict](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
predict(stack.rob)
predict(stack.rob, newdata = stack.dat[c(1,2,4,21), ], se.fit = TRUE)
```

print.covfm	<i>Print Method</i>
-------------	---------------------

Description

The generic print method for objects of class "cov", "covRob", and "covfm".

Usage

```
## S3 method for class 'cov':  
print(x, ...)  
## S3 method for class 'covRob':  
print(x, ...)  
## S3 method for class 'covfm':  
print(x, ...)
```

Arguments

x an object of class "cov", "covRob", or "covfm".
... additional arguments to be passed to the print functions, for example digits.

Value

x is invisibly returned.

Side Effects

a short description of the object is displayed in the console.

See Also

[print](#), [cov](#), [covRob](#), [fit.models](#).

Examples

```
data(woodmod.dat)  
woodmod.cov <- cov(woodmod.dat)  
print(woodmod.cov)  
  
woodmod.covRob <- covRob(woodmod.dat)  
print(woodmod.covRob)  
  
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)  
print(woodmod.fm)
```

```
print.glmRob
```

Print Method for glmRob Objects

Description

Displays the call, coefficients, degrees of freedom, and the residual deviance in a fitted glmRob object.

Usage

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

Arguments

`x` a glmRob object.

`...` additional arguments required by the generic print function. In particular the 'digits' argument is useful for specifying the number of significant digits when displaying numeric output.

Value

`x` is invisibly returned.

See Also

[glmRob](#), [glmRob.object](#).

Examples

```
data(breslow.dat)
bres.rob <- glmRob(sumY ~ Age10 + Base4*Trt, family = poisson(), data = breslow.dat)
print(bres.rob)
print(bres.rob, digits = 3)
```

```
print.glmfm
```

Print Method for glmfm Objects

Description

Displays the calls, coefficient estimates and residual deviance estimates for the glm and/or glmRob objects contained in a glmfm object.

Usage

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

Arguments

`x` a glmfm object.
`...` additional arguments required by the generic print function. In particular the 'digits' argument is useful for specifying the number of significant digits when displaying numeric output.

Value

`x` is invisibly returned.

See Also

[fit.models](#), [glmRob](#), [glm](#), [print](#).

Examples

```
data(breslow.dat)
bres.fm <- fit.models(list(Robust = "glmRob", Classical = "glm"),
                     formula = sumY ~ Age10 + Base4*Trt,
                     family = poisson(), data = breslow.dat)
print(bres.fm)
print(bres.fm, digits = 4)
```

print.lmRob

Print Method for lmRob Objects

Description

Displays the call, coefficients, degrees of freedom, and the residual standard error in a fitted lmRob object.

Usage

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

Arguments

`x` an lmRob object.
`...` additional arguments required by the generic print function. In particular the 'digits' argument is useful for specifying the number of significant digits when displaying numeric output.

Value

`x` is invisibly returned.

See Also

[lmRob](#), [lmRob.object](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
print(stack.rob)
print(stack.rob, digits = 3)
```

print.lmfm

Print Method for lmfm Objects

Description

Prints a brief description of the models contained in an `lmfm` object.

Usage

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

Arguments

`x` an `lmfm` object.
`...` additional arguments required by the generic `print` function.

Value

`x` is invisibly returned.

Examples

```
data(stack.dat)
stack.fm <- fit.models(list(Robust = "lmRob", LS = "lm"), Loss ~ ., data = stack.dat)
print(stack.fm)
```

```
print.summary.covfm
```

Print Method

Description

The generic print method for objects of class "summary.cov", "summary.covRob", and "summary.covfm".

Usage

```
## S3 method for class 'summary.cov':  
print(x, print.distance = TRUE, ...)  
## S3 method for class 'summary.covRob':  
print(x, print.distance = TRUE, ...)  
## S3 method for class 'summary.covfm':  
print(x, print.distance = FALSE, ...)
```

Arguments

`x` an object of class "summary.cov", "summary.covRob", or "summary.covfm".

`print.distance` a logical flag. If `print.distance = TRUE` the mahalanobis distances are printed as well.

`...` additional arguments to be passed to the print functions, for example `digits`.

Value

`x` is invisibly returned.

Side Effects

a description of the summary object is displayed in the console.

See Also

[print](#), [summary](#), [cov](#), [covRob](#), [fit.models](#).

Examples

```
data(woodmod.dat)  
woodmod.cov <- cov(woodmod.dat)  
woodmod.sc <- summary(woodmod.cov)  
print(woodmod.sc)  
  
woodmod.covRob <- covRob(woodmod.dat)  
woodmod.scr <- summary(woodmod.covRob)  
print(woodmod.scr, digits = 4)
```

```
woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
woodmod.sfm <- summary(woodmod.fm)
print(woodmod.sfm, print.distances = TRUE)
```

```
print.summary.glmRob
```

Print Method for summary.glmRob Objects

Description

Displays the call, quartiles of the deviance residuals, an anova table for the coefficients, the dispersion parameter, null deviance, residual deviance, and degrees of freedom. If the `summary.glmRob` object contains a correlation matrix for the coefficients it will be displayed as well.

Usage

```
## S3 method for class 'summary.glmRob':
print(x, ...)
```

Arguments

<code>x</code>	a <code>glmRob</code> object.
<code>...</code>	additional arguments required by the generic <code>print</code> function. In particular the 'digits' argument is useful for specifying the number of significant digits when displaying numeric output.

Value

`x` is invisibly returned.

See Also

[glmRob](#), [glmRob.object](#).

Examples

```
data(breslow.dat)
bres.rob <- glmRob(sumY ~ Age10 + Base4*Trt, family = poisson(), data = breslow.dat)
bres.sum <- summary(bres.rob)
print(bres.sum)
print(bres.sum, digits = 4)
```

```
print.summary.glmfm
```

Print Method for summary.glmfm Objects

Description

Displays a side-by-side comparison of the summaries in a `summary.glmfm` object.

Usage

```
## S3 method for class 'summary.glmfm':  
print(x, ...)
```

Arguments

`x` a `summary.glmfm` object.
`...` additional arguments required by the generic `print` function.

Value

`x` is invisibly returned.

See Also

[summary.glmfm](#).

Examples

```
data(breslow.dat)  
bres.fm <- fit.models(list(Robust = "glmRob", Classical = "glm"),  
                      formula = sumY ~ Age10 + Base4*Trt,  
                      family = poisson(), data = breslow.dat)  
bres.sum <- summary(bres.fm)  
print(bres.sum)  
print(bres.sum, digits = 4)
```

```
print.summary.lmRob
```

Print Method for summary.lmRob Objects

Description

Displays the call, quartiles of the residuals, an anova table for the coefficients, residual standard error, degrees of freedom, robust multiple r-squared, and the test for bias. If the `summary.lmRob` object contains a correlation matrix for the coefficients it will be displayed as well.

Usage

```
## S3 method for class 'summary.lmRob':  
print(x, ...)
```

Arguments

`x` a `summary.lmRob` object.

`...` additional arguments required by the generic `print` function. In particular the 'digits' argument is useful for specifying the number of significant digits when displaying numeric output.

Value

`x` is invisibly returned.

See Also

[lmRob](#), [lmRob.object](#).

Examples

```
data(stack.dat)  
stack.robsum <- summary(lmRob(Loss ~ ., data = stack.dat))  
print(stack.robsum, digits = 4)
```

`print.summary.lmfm` *Print Method for summary.lmfm Objects*

Description

Displays a side-by-side comparison of the summaries in an `lmfm` object.

Usage

```
## S3 method for class 'summary.lmfm':  
print(x, ...)
```

Arguments

`x` a `summary.lmfm` object.

`...` additional arguments required by the generic `print` function.

Value

`x` is invisibly returned.

See Also

[summary.lmf](#).

Examples

```
data(stack.dat)
stack.fm <- fit.models(list(Robust = "lmRob", LS = "lm"), Loss ~ ., data = stack.dat)
stack.sum <- summary(stack.fm)
print.summary.lmf(stack.sum)
print(stack.sum)
```

qqplot.glmRob *Theoretical Quantiles*

Description

Compute theoretical quantiles for the deviance residuals obtained from Poisson and binomial regression models.

Usage

```
qqplot.glmRob(y, par, dist)
```

Arguments

y	a numeric vector containing the response variable.
par	a numeric vector containing the estimated parameter values.
dist	a single integer value specifying the distribution: 0 for binomial; 1 for Poisson.

Value

A list with the following components:

quantiles	a numeric vector containing the theoretical quantiles.
deviance.residuals	a numeric vector containing the deviance residuals.

<code>rb</code>	<i>Robust Bootstrap Standard Errors</i>
-----------------	---

Description

Computes a robust bootstrap estimate of the standard error for each coefficient estimate in a robustly fitted linear model. This function is called by `summary.lmRob` and is not intended to be called directly by users.

Usage

```
rb(lmRob.object, M = 1000, seed = 99, fixed = TRUE)
```

Arguments

`lmRob.object` an `lmRob` object.
`M` a positive integer giving the number of bootstrap subsamples.
`seed` a positive integer specifying the seed for the random number generator.
`fixed` a logical value. This should be set to `TRUE`.

Value

a numeric vector of robust bootstrap standard error estimates.

See Also

[lmRob](#), [summary.lmRob](#).

<code>rockem</code>	<i>Constrained M-estimation</i>
---------------------	---------------------------------

Description

Compute a robust estimate of location and scatter using Rocke's constrained M-estimator.

Usage

```
rockem(x, control)
```

Arguments

`x` a numeric matrix containing the data.
`control` a list of control parameters. The utility function `covRob.control` creates a list of the control parameters and their default values. See details for the required control parameters and their default values.

Details

This function is called by the high-level function `covRob` when the M-estimator is specified (via the optional argument `estim = "m"`). It may also be of interest to power users who want to compute a constrained m-estimate with a minimum of fuss.

The required control parameters and their default values are:

quan = 0.5 an integer value giving the number of observations whose covariance determinant will be minimized.

ntrial = 500 a positive integer specifying the number of random trial subsamples that are drawn for large datasets.

The control parameters `quan` and `ntrial` are passed to the function `fastmcd` which computes the initial robust estimate. The remaining parameters affect the constrained m-estimator.

r = 0.45 a numeric value between 0 and 0.5 giving the desired breakdown point.

alpha = 0.05 a numeric value between 0 and 1 specifying the fraction of points receiving zero weight.

tau = 1e-06 a positive numeric value specifying the tolerance used to determine the singularity of the estimated scatter matrix.

tol = 1e-03 a positive numeric value giving the relative precision for the solution of the M-estimate.

maxit = 150 a positive integer specifying the maximum number of m iterations.

Value

a list with the following components:

<code>call</code>	an image of the call that produced the object with all the arguments named.
<code>cov</code>	a numeric matrix containing the constrained M-estimate of the covariance/correlation matrix.
<code>center</code>	a numeric vector containing the constrained M-estimate of the location vector.
<code>raw.cov</code>	a numeric matrix containing the initial robust estimate of the covariance/correlation matrix.
<code>raw.center</code>	a numeric vector containing the initial robust estimate of the location vector.

References

David M. Rocke (1996). Robustness properties of S-estimators of multivariate location and shape in high dimension. *Annals of Statistics*, Vol. 24, No. 3, 1327-1345.

See Also

[covRob](#), [covRob.control](#).

Examples

```
data(woodmod.dat)
X <- as.matrix(woodmod.dat)
m.control <- covRob.control("m")
rockem(X, m.control)
```

```
s.logistic.misclass.fit
```

S Logistic Misclassification Fitter

Description

Computes the estimate for the consistent misclassification method.

Usage

```
s.logistic.misclass.fit(x, y, mc.gamma, maxit, mc.trc, mc.tol, beta1)
```

Arguments

x	model matrix.
y	response.
mc.gamma	gamma parameter.
maxit	maximum number of iterations.
mc.trc	a logical value. If TRUE diagnostic information is displayed during the computation of the fit.
mc.tol	tolerance.
beta1	initial parameter estimates.

Value

a list containing the parameter estimates.

See Also

[glmRob.misclass](#)

splus.assign *Assign Translator*

Description

Converts the R assign format to the Splus assign format.

Usage

```
splus.assign(asgn, term.labels)
```

Arguments

asgn an assign attribute.
term.labels a term.labels attribute.

Value

an S-Plus like assign list.

stack.dat *Brownlee's Stack-Loss Data*

Description

These data are from the operation of a plant for the oxidation of ammonia to nitric acid, measured on 21 consecutive days.

Usage

```
data(stack.dat)
```

Format

This data frame contains the following variables:

Loss the percentage of ammonia lost (times 10).

Air.Flow air flow into the plant

Water.Temp cooling water inlet temperature.

Acid.Conc. acid concentration as a percentage (coded by subtracting 50 and then multiplying by 10).

Source

Brownlee, K.A. (1965). *Statistical Theory and Methodology in Science and Engineering*. New York: John Wiley & Sons, Inc.

Examples

```
data(stack.dat)
stack.dat
```

```
step.lmRob
```

Build a Model in a Stepwise Fashion

Description

Performs stepwise model selection on a robustly fitted linear model. Presently only the backward stepwise procedure is implemented.

Usage

```
step.lmRob(object, scope, scale, direction = c("both", "backward", "forward"), trace)
```

Arguments

<code>object</code>	an <code>lmRob</code> object.
<code>scope</code>	either a formula or a list with elements <code>lower</code> and <code>upper</code> each of which is a formula. The terms in the right-hand-side of <code>lower</code> are always included in the model and the additional terms in the right-hand-side of <code>upper</code> are the candidates for inclusion/exclusion from the model. If a single formula is given, it is taken to be <code>upper</code> and <code>lower</code> is set to the empty model. The <code>.</code> operator is interpreted in the context of the formula in <code>object</code> .
<code>scale</code>	a single numeric value containing a residual scale estimate. If missing, the scale estimate in <code>object</code> is used.
<code>direction</code>	a character value specifying the mode of stepwise search. The possibilities are "both", "backward", and "forward", with a default of "backward". Presently only "backward" stepwise searches are implemented.
<code>trace</code>	a logical value. If <code>TRUE</code> , information is printed during stepwise search.
<code>keep</code>	a filter function whose input is a fitted model object and the associated AIC statistic, and whose output is arbitrary. Typically <code>keep</code> will select a subset of the components of the object and return them. The default is not to keep anything.
<code>steps</code>	an integer value specifying the the maximum number of steps to be considered. The default is 1000 (essentially as many as required). It is typically used to stop the process early.
<code>fast</code>	a logical value. If <code>TRUE</code> the robust initial estimate (used when fitting each of the reduced models) is replaced by a weighted least squares estimate using the robust weights computed for the current fit.
<code>...</code>	additional arguments required by the generic step function.

Details

Presently only backward stepwise selection is supported. During each step the Robust Final Prediction Error (as computed by the function `lmRob.RFPE`) is calculated for the current model and for each sub-model achievable by deleting a single term. The function then either steps to the sub-model with the lowest Robust Final Prediction Error or, if the current model has the lowest Robust Final Prediction Error, terminates. The scale estimate from `object` is used to compute the Robust Final Prediction Error throughout the procedure unless the `scale` argument is provided in which case the user specified value is used.

Value

the model with the lowest Robust Final Prediction Error encountered during the stepwise procedure is returned. Additionally, an `anova` element corresponding to the steps taken in the search is appended to the returned object. If a `keep` function was provided then the kept values can be found in the `keep` element of the returned object.

See Also

`/code`[lmRob](#), `/code`[lmRob.RFPE](#), `/code`[add1.lmRob](#), `/code`[drop1.lmRob](#).

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)

## The default behavior is to try dropping all terms ##
step.lmRob(stack.rob)

## Keep Water.Temp in the model ##
my.scope <- list(lower = . ~ Water.Temp, upper = . ~ .)
step.lmRob(stack.rob, scope = my.scope)

## Try the fast procedure ##
step.lmRob(stack.rob, scope = my.scope, fast = TRUE)
```

`stop.on.bdObject` *Stop on Big Data Object*

Description

R doesn't have big data objects so this function does nothing.

Usage

```
stop.on.bdObject(x)
```

Arguments

`x` a `data.frame`.

Value

NULL

summary.covfm	<i>Summary Method</i>
---------------	-----------------------

Description

The generic summary method for objects of class "cov", "covRob", and "covfm".

Usage

```
## S3 method for class 'cov':
summary(object, ...)
## S3 method for class 'covRob':
summary(object, ...)
## S3 method for class 'covfm':
summary(object, ...)
```

Arguments

object	an object of class "cov", "covRob", or "covfm".
...	additional arguments for the summary method.

Value

an object of class "summary.cov", "summary.covRob", or "summary.covfm" respectively. Objects of class "summary.cov" and "summary.covRob" have the following components. Objects of class "summary.covfm" are lists whose elements are "summary.cov" and "summary.covRob" objects.

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

See Also

[summary](#), [cov](#), [covRob](#), [fit.models](#).

Examples

```

data(woodmod.dat)
woodmod.cov <- cov(woodmod.dat)
woodmod.sc <- summary(woodmod.cov)

woodmod.covRob <- covRob(woodmod.dat)
woodmod.scr <- summary(woodmod.covRob)

woodmod.fm <- fit.models(list(Robust = "covRob", Classical = "cov"), data = woodmod.dat)
woodmod.sfm <- summary(woodmod.fm)

```

summary.glmRob

*Summarizing Robust Generalized Linear Model Fits***Description**

Compute a summary of the robustly fitted generalized linear model.

Usage

```

## S3 method for class 'glmRob':
summary(object, correlation = TRUE, ...)

```

Arguments

object	a glmRob object.
correlation	a logical value. If TRUE then the correlation matrix of the coefficients is included in the summary.
...	additional arguments required by the generic summary function.

Value

The summary is returned in a list of class summary.glmRob and contains the following components:

comp1	Description of 'comp1'
comp2	Description of 'comp2'
...	

Examples

```

data(breslow.dat)
bres.rob <- glmRob(sumY ~ Age10 + Base4*Trt, family = poisson(), data = breslow.dat)
bres.sum <- summary(bres.rob)
bres.sum

```

```
summary.glmfm      Summary Method for glmfm Objects
```

Description

Compute a summary of each glm or glmRob model in a glmfm object.

Usage

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

Arguments

```
object      a glmfm object.
correlation a logical value. If TRUE, correlation matrices of the coefficient estimates are
            included in each summary.
...         additional arguments required by the generic summary function.
```

Value

a list with class summary.glmfm whose elements are summaries of each model in object.

See Also

[fit.models](#), [summary](#), [summary.glmRob](#), [summary.glm](#).

Examples

```
data(breslow.dat)
bres.fm <- fit.models(list(Robust = "glmRob", Classical = "glm"),
                     formula = sumY ~ Age10 + Base4*Trt,
                     family = poisson(), data = breslow.dat)
bres.sum <- summary(bres.fm)
bres.sum
```

```
summary.lmRob      Summarizing Robust Linear Model Fits
```

Description

Compute a summary of the robustly fitted linear model.

Usage

```
## S3 method for class 'lmRob':
summary(object, correlation = FALSE, bootstrap.se = FALSE, ...)
```

Arguments

object	an lmRob object.
correlation	a logical value. If TRUE then the correlation matrix of the coefficients is included in the summary.
bootstrap.se	a logical value. If TRUE then bootstrap standard error estimates are included in the summary.
...	additional arguments required by the generic summary function.

Value

The summary is returned in a list of class summary.lmRob and contains the following components:

sigma	a single numeric value containing the residual scale estimate.
df	a numeric vector of length 3 containing integer values: the rank of the model matrix, the residual degrees of freedom, and the number of coefficients in the model.
cov.unscaled	the unscaled covariance matrix; i.e, the matrix that, when multiplied by the estimate of the error variance, yields the estimated covariance matrix for the coefficients.
correlation	the correlation coefficient matrix for the coefficients in the model.
...	the remaining components are the same as the corresponding components in an lmRob object. Use the names function to obtain a list of the components.

Examples

```
data(stack.dat)
stack.rob <- lmRob(Loss ~ ., data = stack.dat)
stack.sum <- summary(stack.rob)
stack.sum
stack.bse <- summary(stack.rob, bootstrap.se = TRUE)
stack.bse
```

summary.lmfm

Summary Method for lmfm Objects

Description

Compute a summary of each lm or lmRob model in a summary.lmfm object.

Usage

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

Arguments

`object` an `lmfm` object.

`correlation` a logical value. If `TRUE`, the correlation matrices for the coefficients are included in the summaries.

`...` additional arguments required by the generic `summary` function.

Value

a list with class `summary.lmfm` whose elements are summaries of each model in `object`.

See Also

`fit.models`, `summary`, `summary.lmRob`, `summary.lm`.

Examples

```
data(stack.dat)
stack.fm <- fit.models(list(Robust = "lmRob", LS = "lm"), Loss ~ ., data = stack.dat)
stack.sum <- summary(stack.fm)
stack.sum
```

test.lmRob

Various Tests of Robust Regression Estimates

Description

Conducts test for bias of robust MM-estimates and Least Squares (LS) estimates against S-estimates, or permutation test of the slope estimate in a straight line fit.

Usage

```
test.lmRob(object, type = "bias", level = NULL, n.permute = 99)
```

Arguments

`object` an object of class `"lmRob"`.

`type` character string. Valid choices are `"bias"` for bias test, or `"permutation"` for permutation test.

`level` the level of the test for bias of MM-estimate. By default, the `level` component of `object$robust.control` is used.

`n.permute` a positive integer value specifying the number of permutations to use.

Value

the p-value of the permutation test, or an object of class "biasMM" representing the bias test, in which case the following components ARE included:

mm	a list describing the test of bias for final MM-estimates, with the following components: <code>stat</code> , the t-statistic; <code>pchi</code> , a chi-squared p-value; <code>qchi</code> , the quantile of the chi-squared distribution with degrees of freedom equal to <code>object\$rank</code> corresponding to the probability input in the level (<code>object\$robust.control\$level</code>).
ls	a list describing the test of bias for LS-estimates, with the following components: <code>stat</code> , the t-statistic; <code>pchi</code> , a chi-squared p-value.
level	the level of the test for bias of MM-estimate.

References

Yohai, V., Stahel, W. A., and Zamar, R. H. (1991). A procedure for robust estimation and inference in linear regression, in Stahel, W. A. and Weisberg, S. W., Eds., *Directions in robust statistics and diagnostics, Part II*. Springer-Verlag.

undocumented.glmRob

Undocumented glmRob Helper Functions

Description

These functions are used by `glmRob` and its methods. They are not intended to be called directly by users and are not guaranteed to be included in future versions of the Robust Library.

Usage

```
glmRob.gfedca(vtheta, ci, wa, ni, oi = 0, ics)
glmRob.gicstp(ics, ialg, ni, vtheta, ai, oi, tol, maxit)
glmRob.gintac(x, y, ni, oi, ics, maxtt, maxta, tolt, tola, b, c)
glmRob.glmdev(y, ni, ci, wa, vtheta, offset = 0, ics = ics)
glmRob.gymain(x, y, ni, cov, a, theta, oi = 0, b,
  gam, tau, ics, iugl, iopt, ialg, icnvt, icnva, maxit, maxtt,
  maxta, maxtc, tol, tolt, tola, tolc, trc = FALSE)
glmRob.gytstp(x, y, ci, theta, wa, cov, ni, offset, tol, ics, maxit)
glmRob.ktaskw(x, d, e, tau, ia, f, fl, iainv, a)
```

Arguments

vtheta	vtheta
ci	ci
wa	wa
ni	ni

oi	oi
icase	icase
ialg	ialg
ai	ai
tol	tol
maxit	maxit
x	x
y	y
maxtt	maxtt
maxta	maxta
tolt	tolt
tola	tola
b	b
c	c
offset	offset
cov	cov
a	a
theta	theta
gam	gam
tau	tau
iugl	iugl
iopt	iopt
icnvt	icnvt
icnva	icnva
maxtc	maxtc
tolc	tolc
trc	trc
d	d
e	e
ia	ia
f	f
f1	f1
iainv	iainv

See Also

[glmRob](#)

undocumented.lmRob *Undocumented lmRob Helper Functions*

Description

These functions are used by `lmRob` and its methods. They are not intended to be called directly by users and are not guaranteed to be included in future versions of the Robust Library.

Usage

```
lmRob.fit(x, y, x1 = NULL, x2 = NULL, x1.idx = NULL, nrep = NULL, robust.control =
lmRob.wfit(x, y, w, x1 = NULL, x2 = NULL, x1.idx = NULL, nrep = NULL, robust.control
lmRob.const(eff, ipsi = 1)
lmRob.eff0(mu = 1, itype = 1, iucv = 1, iwww = 1, ialfa = 1, sigmx = 1, upper = 10,
lmRob.effad(eff)
lmRob.lar(x, y, tol = 1e-06)
lmRob.wm(x, y, coeff0, ucov0, scale0, itype = 1, isigma = -1, ipsi = 1, xk = 0.9440
```

Arguments

<code>x</code>	<code>x</code>
<code>y</code>	<code>y</code>
<code>x1</code>	<code>x1</code>
<code>x2</code>	<code>x2</code>
<code>x1.idx</code>	<code>x1.idx</code>
<code>nrep</code>	<code>nrep</code>
<code>robust.control</code>	<code>robust.control</code>
<code>genetic.control</code>	<code>genetic.control</code>
<code>...</code>	<code>...</code>
<code>w</code>	<code>w</code>
<code>eff</code>	<code>eff</code>
<code>ipsi</code>	<code>ipsi</code>
<code>mu</code>	<code>mu</code>
<code>itype</code>	<code>itype</code>
<code>iucv</code>	<code>iucv</code>
<code>iwww</code>	<code>iwww</code>
<code>ialfa</code>	<code>ialfa</code>
<code>sigmx</code>	<code>sigmx</code>
<code>upper</code>	<code>upper</code>

til	til
maxit	maxit
tol	tol
epmach	epmach
uflow	uflow
ta	ta
tb	tb
tc	tc
coeff0	coeff0
ucov0	ucov0
scale0	scale0
isigma	isigma
xk	xk
beta	beta
wgt	wgt
tlo	tlo
tua	tua
mxr	mxr

See Also

[lmRob](#)

update.lmRob	<i>Update an lmRob Model Object</i>
--------------	-------------------------------------

Description

This is a method for the function [update](#) for objects inheriting from class `lmRob`. See [update](#) for the general behavior of this function and for the interpretation of the arguments.

Usage

```
## S3 method for class 'lmRob':  
update(object, formula, evaluate = TRUE, class, ...)
```

Arguments

object	an lmRob object.
formula	a modeling formula, such as $y \sim a + b$. A single dot <code>.</code> on either side of the <code>~</code> gets replaced by the left or right side of the formula in <code>object</code> . The dot on the left can be omitted. By default, it refits <code>object</code> using the same formula as in <code>object</code> .
evaluate	a logical value. If <code>TRUE</code> the updated call is evaluated and returned. Otherwise the unevaluated call is returned.
class	a single character value specifying the fitting class for the new object.
...	additional arguments passed to the generic update function.

Details

If `formula` is missing, `update.lmRob` alternates between the initial estimates and final estimates. If `formula` is present, `update.lmRob` functions just like `update.default`.

Value

either a new updated object, or else an unevaluated expression for creating such an object.

See Also

[lmRob.update](#).

weight.funs.q

Weight Functions

Description

These functions compute the weights used by `lmRob` and its associated methods.

Usage

```
psi.weight(svals, ips = 1, xk = 1.06)
rho.weight(svals, ips = 1, xk = 1.06)
psp.weight(svals, ips = 1, xk = 1.06)
chi.weight(svals, ips = 1, xk = 1.06)
```

Arguments

svals	a numeric vector.
ips	choose your weight function: <code>ips = 1</code> for optimal function, <code>ips = 2</code> for rescaled bisquare function, and <code>ips = 3</code> for Huber function.
xk	a numeric value specifying the tuning constant.

Details

See the Theoretical Details section in chapter 2 of Robust.pdf.

Value

a numeric vector containing the weights.

woodmod.dat

Modified Wood Data

Description

The explanatory variables from the Modified Data on Wood Specific Gravity analyzed in Rousseeuw and Leroy (1987).

Usage

```
data(woodmod.dat)
```

Format

This data frame contains the following variables:

- V1** number of fibers per square milimeter in Springwood (coded by dividing by 1000).
- V2** number of fibers per square milimeter in Summerwood (coded by dividing by 10000).
- V3** fraction of Springwood.
- V4** fraction of light absorption by Springwood.
- V5** fraction of light absorption by Summerwood.

Source

Rousseeuw, P. J., and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. New York: Wiley.

Examples

```
data(woodmod.dat)  
woodmod.dat
```

`wt.carroll`*Weight Functions*

Description

Compute various weights.

Usage

```
wt.carroll(u, b)
wt.huber(u, const = 1.345)
```

Arguments

<code>u</code>	a numeric vector.
<code>b</code>	a single numeric value specifying the parameter b.
<code>const</code>	a tuning constant.

Value

a numeric vector containing the weights.

Index

*Topic **aplot**

- key, 35
- panel.addons, 53
- qqplot.glmRob, 70

*Topic **datasets**

- breslow.dat, 5
- leuk.dat, 36
- mallows.dat, 50
- stack.dat, 74
- woodmod.dat, 87

*Topic **hplot**

- covfmDistance2Plot, 11
- covfmEllipsesPlot, 12
- covfmScreePlot, 13
- covfmSqrtMDPlot, 14
- lmfm2DRegPlot, 44
- lmfmOverlaidQQPlot, 45
- lmfmOverlaidResDenPlot, 45
- lmfmResKernDenPlot, 46
- lmfmRespVsFittedPlot, 48
- lmfmResQQPlot, 46
- lmfmResVsFittedPlot, 47
- lmfmSqrtResVsFittedPlot, 49
- lmfmSRvsRDPlot, 48
- lmfmStdResPlot, 50
- plot.covfm, 54
- plot.glmfm, 56
- plot.glmRob, 55
- plot.lmf, 58
- plot.lmRob, 57

*Topic **methods**

- add1.lmRob, 2
- anova.glmRob, 3
- anova.lmRob, 4
- coef.glmfm, 6
- coef.lmf, 7
- drop1.lmRob, 16
- lmRob.object, 42
- methods.glmRob, 51

- methods.lmRob, 52
- plot.covfm, 54
- plot.glmfm, 56
- plot.glmRob, 55
- plot.lmf, 58
- plot.lmRob, 57
- predict.glmRob, 59
- predict.lmRob, 61
- print.covfm, 62
- print.glmfm, 63
- print.glmRob, 63
- print.lmf, 65
- print.lmRob, 64
- print.summary.covfm, 66
- print.summary.glmfm, 68
- print.summary.glmRob, 67
- print.summary.lmf, 69
- print.summary.lmRob, 68
- step.lmRob, 75
- summary.covfm, 77
- summary.glmfm, 79
- summary.glmRob, 78
- summary.lmf, 80
- summary.lmRob, 79
- update.lmRob, 85

*Topic **models**

- fit.models, 21
- glmRob, 24
- glmRob.object, 33
- lmRob, 37
- lmRob.fit.compute, 41
- lmRob.object, 42

*Topic **multivariate**

- cov, 8
- covRob, 9
- donostah, 15
- fastcov, 17
- fastmcd, 19
- rockem, 71

***Topic regression**

add1.lmRob, 2
 anova.glmRob, 3
 anova.lmRob, 4
 drop1.lmRob, 16
 glmRob, 24
 glmRob.control, 26
 glmRob.cubif, 27
 glmRob.cubif.control, 28
 glmRob.cubif.Ini, 27
 glmRob.cubif.null, 29
 glmRob.Initial.LMS, 24
 glmRob.mallows, 30
 glmRob.mallows.control, 31
 glmRob.misclass, 31
 glmRob.misclass.control, 32
 glmRob.misclass.w, 33
 glmRob.object, 33
 lmRob, 37
 lmRob.control, 39
 lmRob.fit.compute, 41
 lmRob.object, 42
 lmRob.RFPE, 36
 predict.glmRob, 59
 predict.lmRob, 61
 rb, 71
 s.logistic.misclass.fit, 73
 step.lmRob, 75
 summary.glmfm, 79
 summary.glmRob, 78
 summary.lmfm, 80
 summary.lmRob, 79
 test.lmRob, 81
 update.lmRob, 85

***Topic robust**

add1.lmRob, 2
 anova.glmRob, 3
 anova.lmRob, 4
 covRob, 9
 donostah, 15
 drop1.lmRob, 16
 fastcov, 17
 fastmcd, 19
 glmRob, 24
 glmRob.control, 26
 glmRob.cubif, 27
 glmRob.cubif.control, 28
 glmRob.cubif.Ini, 27

glmRob.cubif.null, 29
 glmRob.Initial.LMS, 24
 glmRob.mallows, 30
 glmRob.mallows.control, 31
 glmRob.misclass, 31
 glmRob.misclass.control, 32
 glmRob.misclass.w, 33
 glmRob.object, 33
 lmRob, 37
 lmRob.control, 39
 lmRob.effvy, 40
 lmRob.fit.compute, 41
 lmRob.ga, 42
 lmRob.object, 42
 lmRob.RFPE, 36
 lmRob.ucovcoef, 44
 predict.glmRob, 59
 predict.lmRob, 61
 rb, 71
 rockem, 71
 s.logistic.misclass.fit, 73
 step.lmRob, 75
 summary.glmfm, 79
 summary.glmRob, 78
 summary.lmfm, 80
 summary.lmRob, 79
 test.lmRob, 81
 undocumented.glmRob, 82
 undocumented.lmRob, 84
 update.lmRob, 85
 weight.funs.q, 86
 wt.carroll, 88

***Topic utilities**

.First.lib, 1
 arg.names, 5
 covRob.control, 10
 gen.data, 23
 get.fit.models.database, 23
 splus.assign, 74
 stop.on.bdObject, 76
 .First.lib, 1

add1, 2, 3, 17, 43
 add1.lmRob, 2, 37, 76
 anova, 3, 4, 17, 43
 anova.glmRob, 3
 anova.glmRoblist, 3
 anova.glmRoblist (anova.glmRob), 3
 anova.lmRob, 4

- anova.lmRoblist, 4
- anova.lmRoblist (*anova.lmRob*), 4
- arg.names, 5
- breslow.dat, 5
- chi.weight (*weight.funs.q*), 86
- coef, 6, 7, 43
- coef.glmfm, 6
- coef.glmRob (*methods.glmRob*), 51
- coef.lmfm, 7
- coef.lmRob (*methods.lmRob*), 52
- cov, 8, 10, 12–14, 55, 62, 66, 77
- cov.wt, 9
- covfmDistance2Plot, 11, 55
- covfmEllipsesPlot, 12, 55
- covfmScreePlot, 13, 55
- covfmSqrtMDPlot, 14, 55
- covRob, 9, 9, 11–16, 18, 21, 55, 62, 66, 72, 77
- covRob.control, 10, 10, 15, 16, 18, 72
- deviance, 43
- deviance.lmRob (*methods.lmRob*), 52
- donostah, 10, 11, 15
- drop1, 3, 17, 43
- drop1.lmRob, 16, 37, 76
- family.glmRob (*methods.glmRob*), 51
- fastcov, 10, 11, 17
- fastmcd, 10, 11, 19, 72
- fit.models, 12–14, 21, 23, 55, 57, 59, 62, 64, 66, 77, 79, 81
- fitted, 43
- formula, 43
- formula.lmRob (*methods.lmRob*), 52
- gen.data, 23
- get.fit.models.database, 22, 23
- glm, 26, 64
- glmRob, 3, 24, 27–29, 31–33, 35, 56, 60, 63, 64, 67, 82, 83
- glmRob.control, 26, 26
- glmRob.cubif, 24, 27, 27
- glmRob.cubif.control, 26, 28, 28
- glmRob.cubif.Ini, 27
- glmRob.cubif.null, 29
- glmRob.gfedca
(*undocumented.glmRob*), 82
- glmRob.gicstp
(*undocumented.glmRob*), 82
- glmRob.gintac
(*undocumented.glmRob*), 82
- glmRob.glmdev
(*undocumented.glmRob*), 82
- glmRob.gymain
(*undocumented.glmRob*), 82
- glmRob.gytstp
(*undocumented.glmRob*), 82
- glmRob.Initial.LMS, 24
- glmRob.ktaskw
(*undocumented.glmRob*), 82
- glmRob.mallows, 30
- glmRob.mallows.control, 26, 31
- glmRob.misclass, 31, 33, 73
- glmRob.misclass.control, 26, 32, 33
- glmRob.misclass.F
(*glmRob.misclass.w*), 33
- glmRob.misclass.f
(*glmRob.misclass.w*), 33
- glmRob.misclass.G
(*glmRob.misclass.w*), 33
- glmRob.misclass.g
(*glmRob.misclass.w*), 33
- glmRob.misclass.w, 33
- glmRob.object, 25, 26, 28, 30, 32, 33, 63, 67
- key, 35
- labels, 43
- labels.glmRob (*methods.glmRob*), 51
- labels.lmRob (*methods.lmRob*), 52
- leuk.dat, 36
- lmfm2DRegPlot, 44, 59
- lmfmOverlaidQQPlot, 45, 59
- lmfmOverlaidResDenPlot, 45, 59
- lmfmResKernDenPlot, 46, 59
- lmfmRespVsFittedPlot, 48, 57, 59
- lmfmResQQPlot, 46, 57, 59
- lmfmResVsFittedPlot, 47, 57, 59
- lmfmSqrtResVsFittedPlot, 49, 57, 59
- lmfmSRvsRDPlot, 48, 57, 59
- lmfmStdResPlot, 50, 57, 59
- lmRob, 3, 4, 37, 37, 40–43, 47, 51–53, 58, 61, 65, 69, 71, 76, 84–86
- lmRob.const, 4

- lmRob.const (*undocumented.lmRob*),
84
- lmRob.control, 38, 39, 39, 42
- lmRob.eff0 (*undocumented.lmRob*),
84
- lmRob.effad (*undocumented.lmRob*),
84
- lmRob.effvy, 4, 40
- lmRob.fit (*undocumented.lmRob*), 84
- lmRob.fit.compute, 41
- lmRob.ga, 42
- lmRob.lar (*undocumented.lmRob*), 84
- lmRob.object, 3, 17, 38, 39, 41, 42, 52, 53,
65, 69
- lmRob.RFPE, 36, 76
- lmRob.ucovcoef, 44
- lmRob.wfit (*undocumented.lmRob*),
84
- lmRob.wm (*undocumented.lmRob*), 84
- loess, 47, 49

- mallows.dat, 50
- methods.glmRob, 51
- methods.lmRob, 52
- model.frame.glmRob
(*methods.glmRob*), 51
- model.frame.lmRob
(*methods.lmRob*), 52
- model.matrix.glmRob
(*methods.glmRob*), 51
- model.matrix.lmRob
(*methods.lmRob*), 52

- names, 80

- panel.addons, 53
- plot, 43, 55–59
- plot.cov (*plot.covfm*), 54
- plot.covfm, 11–14, 54
- plot.covRob (*plot.covfm*), 54
- plot.glmfm, 56, 56
- plot.glmRob, 55
- plot.lmf, 58, 58
- plot.lmRob, 57
- predict, 60, 61
- predict.glmRob, 59
- predict.lmRob, 61
- print, 43, 62, 64–66, 68, 69
- print.cov (*print.covfm*), 62
- print.covfm, 62
- print.covRob (*print.covfm*), 62
- print.glmfm, 63
- print.glmRob, 63
- print.lmf, 65
- print.lmRob, 64
- print.summary.cov
(*print.summary.covfm*), 66
- print.summary.covfm, 66
- print.summary.covRob
(*print.summary.covfm*), 66
- print.summary.glmfm, 68
- print.summary.glmRob, 67
- print.summary.lmf, 69
- print.summary.lmRob, 68
- psi.weight (*weight.funs.q*), 86
- psp.weight (*weight.funs.q*), 86

- qqplot.glmRob, 70

- rb, 71
- residuals, 43
- residuals.glmRob
(*methods.glmRob*), 51
- residuals.lmRob (*methods.lmRob*),
52
- rho.weight (*weight.funs.q*), 86
- rockem, 10, 11, 71

- s.logistic.misclass.fit, 73
- splus.assign, 74
- stack.dat, 74
- step.lmRob, 37, 75
- stop.on.bdObject, 76
- summary, 43, 66, 77–81
- summary.cov (*summary.covfm*), 77
- summary.covfm, 77
- summary.covRob (*summary.covfm*), 77
- summary.glm, 79
- summary.glmfm, 68, 79
- summary.glmRob, 78, 79
- summary.lm, 81
- summary.lmf, 70, 80
- summary.lmRob, 71, 79, 81

- test.lmRob, 81

- undocumented.glmRob, 82
- undocumented.lmRob, 84

update, [43](#), [85](#), [86](#)
update.lmRob, [85](#)

var, [9](#)

weight.funs.q, [86](#)
weights.lmRob(*methods.lmRob*), [52](#)
woodmod.dat, [87](#)
wt.carroll, [88](#)
wt.huber(*wt.carroll*), [88](#)