Package 'marg'

July 22, 2025

Version 1.2-4

Date 2025-05-22

Title Approximate Marginal Inference for Regression-Scale Models

Maintainer Alessandra R. Brazzale <alessandra.brazzale@unipd.it>

Depends R (>= 3.0.0), statmod, survival

Suggests boot, cond, csampling, nlreg

Description Implements likelihood inference based on higher order approximations for linear nonnormal regression models.

License GPL (>= 2) | file LICENCE

URL https://www.r-project.org

LazyLoad yes

LazyData yes

NeedsCompilation no

Author Alessandra R. Brazzale [aut, cre] (author of original code for S, author of R port following earlier work by Douglas Bates)

Repository CRAN

Date/Publication 2025-05-26 16:50:02 UTC

Contents

`marg-package` *Approximate marginal inference for regression-scale models*

Description

Likelihood inference based on higher order approximations for linear nonnormal regression models

Details

Package:	marg
Version:	1.2-0
Date:	2009-10-03
Depends:	R (>= 2.6.0), statmod, survival
Suggests:	boot, cond, csampling, nlreg
License:	GPL (>= 2)
URL:	http://www.r-project.org, http://statwww.epfl.ch/AA/
LazyLoad:	yes
LazyData:	yes

Index:

```
Functions:
=========
cond                    Approximate Conditional Inference - Generic
                        Function
cond.rsm                Approximate Conditional Inference in
                        Regression-Scale Models
```

dHuber	Huber's Least Favourable Distribution
family.rsm	Use family() on a "rsm" object
family.rsm.object	Family Object for Regression-Scale Models
logLik.rsm	Compute the Log Likelihood for Regression-Scale Models
marg.object	Approximate Marginal Inference Object
plot.marg	Generate Plots for an Approximate Marginal Inference Object
print.summary.marg	Use print() on a "summary.marg" object
residuals.rsm	Compute Residuals for Regression-Scale Models
rsm	Fit a Regression-Scale Model
rsm.diag	Diagnostics for Regression-Scale Models
rsm.diag.plots	Diagnostic Plots for Regression-Scale Models
rsm.families	Generate a RSM Family Object
rsm.fit	Fit a Regression-Scale Model Without Computing the Model Matrix
rsm.null	Fit an Empty Regression-Scale Model
rsm.object	Regression-Scale Model Object
rsm.surv	Fit a Regression-Scale Model Without Computing the Model Matrix
summary.marg	Summary Method for Objects of Class "marg"
summary.rsm	Summary Method for Regression-Scale Models
update.rsm	Update and Re-fit a RSM Model Call
vcov.rsm	Calculate Variance-Covariance Matrix for a Fitted RSM Model

Datasets:

darwin	Darwin's Data on Growth Rates of Plants
houses	House Price Data
nuclear	Nuclear Power Station Data
venice	Sea Level Data

Further information is available in the following vignettes:

Rnews-paper	hoa: An R Package Bundle for Higher Order Likelihood Inference (source, pdf)

Author(s)

S original by Alessandra R. Brazzale <alessandra.brazzale@unimore.it>. R port by Alessandra R. Brazzale <alessandra.brazzale@unimore.it>, following earlier work by Douglas Bates.

Maintainer: Alessandra R. Brazzale <alessandra.brazzale@unimore.it>

cond *Approximate Conditional Inference - Generic Function*

Description

Performs approximate conditional inference.

Usage

```
cond(object, offset, ...)
```

Arguments

`object`	a fitted model object. Families supported are binomial and Poisson with canonical link function (class `glm`), and regression-scale models (class `rsm`).
`offset`	the covariate occurring in the model formula whose coefficient represents the parameter of interest. May be numerical or a two-level factor. In case of a two-level factor, it must be coded by contrasts and not appear as two dummy variables in the model. Can also be a call to a mathematical function (such as `exp`, `sin`, `...`) or to a mathematical operator (`\^`, `/`, `...`) applied to a numerical variable. The call must always agree with the label used to identify the corresponding parameter in the fitted model object passed through the `object` argument. Beware that the label includes the identity function `I()` if an arithmetic operator was used. Other function types (e.g. `factor`) and interactions are not admitted.
`...`	absorbs any additional arguments. See `cond.glm` and `cond.rsm` for details.

Details

This function is generic (see `methods`); method functions can be written to handle specific classes of data. Classes which already have methods for this function include: `glm` and `rsm`.

Value

The returned value is an *approximate conditional inference* object. Classes already supported are `cond` and `marg` depending on whether the fitted model object passed through the `object` argument has class `glm` or `rsm`. See `cond.object` or `marg.object` for more details.

References

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne. Chapter 6.

See Also

`cond.glm`, `cond.rsm`, `cond.object`, `marg.object`

Examples

```
## Urine Data
## Not run:
data(urine)
urine.glm <- glm(r ~ gravity + ph + osmo + cond + urea + log(calc),
                 family = binomial, data = urine)
##
## function call as offset variable
labels(coef(urine.glm))
cond(urine.glm, log(calc))
##
## large estimate of regression coefficient
urine.glm <- glm(r ~ gravity + ph + osmo + cond + urea + calc,
                 family = binomial, data = urine)
coef(urine.glm)
urine.glm <- glm(r ~ I(gravity * 100) + ph + osmo + cond + urea + calc,
                 family = binomial, data = urine)
coef(urine.glm)
urine.cond <- cond(urine.glm, I(gravity * 100))
plot(urine.cond, which = 4)

## End(Not run)

## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
##
## parameter of interest: scale parameter
houses.marg <- cond(houses.rsm, scale)
plot(houses.marg, which = 2)
```

`cond.rsm` *Approximate Conditional Inference in Regression-Scale Models*

Description

Performs approximate conditional inference on a scalar parameter of interest in regression-scale models. The output is stored in an object of class `marg`.

Usage

```
## S3 method for class 'rsm'
cond(object, offset, formula = NULL, family = NULL,
     dispersion = NULL, data = sys.frame(sys.parent()), pts = 20,
     n = max(100, 2*pts), tms = 0.6, from = NULL, to = NULL,
     control = glm.control(...), trace = FALSE, ...)
```

Arguments

object	a rsm object; for instance the result of a call to rsm.
offset	either the covariate occurring in the model formula whose coefficient represents the parameter of interest or scale if the parameter of interest is the scale parameter. In the first case, the variable may be numerical or a two-level factor. In case of a two-level factor, it must be coded by contrasts and not as two dummy variables. Can also be a call to a mathematical function (such as exp, sin, ...) or to a mathematical operator (^, /, ...) applied to a numerical variable. The call must always agree with the label used to identify the corresponding parameter in the rsm object passed through the object argument or defined by formula and family. Beware that the label includes the identity function I() if an arithmetic operator was used. Other function types (e.g. factor) and interactions are not admitted. If interest focuses on the scale parameter, it must not be fixed in object or when using the dispersion argument in case no rsm object is supplied.
formula	a formula expression (only if no rsm object is defined).
family	a family.rsm object defining the error distribution (only if no rsm object is defined). See rsm.families for the families supported.
dispersion	argument only to be used if no rsm object is defined. If NULL, the scale parameter is taken to be unknown. If known, the numerical value can be passed. The default is NULL. Huber's least favourable error distribution represents a special case. If dispersion is NULL, the maximum likelihood estimate is computed, while if TRUE the MAD estimate is calculated and the scale parameter fixed to this value in subsequent computations.
data	an optional data frame in which to interpret the variables occurring in the formula (only if no rsm object is defined).
pts	number of output points (minimum 10) that are calculated exactly; the default is 20.
n	approximate number of output points (minimum 50) produced by the spline interpolation. The default is the maximum between 100 and twice pts.
tms	defines the range MLE +/- tms * S.E. where cubic spline interpolation is replaced by polynomial interpolation. The default is 0.6.
from	starting value of the sequence that contains the values of the parameter of interest for which output points are calculated exactly. The default is MLE - 3.5 * S.E.
to	ending value of the sequence that contains the values of the parameter of interest for which output points are calculated exactly. The default is MLE + 3.5 * S.E.
control	a list of iteration and algorithmic constants that control the rsm fit. See glm.control for their names and default values.
trace	if TRUE, iteration numbers will be printed.
...	additional arguments, such as weights, subset, control etc. used by the rsm fitting routine if the rsm object is defined through formula and family. See rsm for their definition and use.

Details

This function is a method for the generic function cond for class rsm. It can be invoked by calling cond for an object of the appropriate class, or directly by calling cond.rsm regardless of the class of the object. cond.rsm has also to be used if the rsm object is not provided throught the object argument but specified by formula and family.

The function cond.rsm implements several small sample asymptotic methods for approximate conditional inference in regression-scale models. Approximations for both the modified/marginal log likelihood function and approximate conditional/marginal tail probabilities are available (see marg.object for details). Attention is restricted to a scalar parameter of interest, either a regression coefficient or the scale parameter. In the first case, the associated covariate may be either numerical or a two-level factor.

Approximate conditional (or equivalently marginal) inference is performed by either updating a fitted regression-scale model or defining the model formula and family. All approximations are calculated exactly for pts equally spaced points ranging from from to to. A spline interpolation is used to extend them over the whole interval of interest, except for the range of values defined by MLE +/- tms * S.E. where the spline interpolation is replaced by a higher order polynomial interpolation. This is done in order to avoid numerical instabilities which are likely to occur for values of the parameter of interest close to the MLE. Results are stored in an object of class marg. Method functions like print, summary and plot can be used to examine the output or represent it graphically. Components can be extracted using coef, formula and family.

Main references for the methods considered are the papers by *Barndorff-Nielsen (1991)*, *DiCiccio, Field and Fraser (1990)* and *DiCiccio and Field (1991)*. The theory and statistics used are summarized in *Brazzale (2000, Chapters 2 and 3)*. More details of the implementation are given in *Brazzale (1999; 2000, Section 6.3.1)*.

Value

The returned value is an object of class marg; see marg.object for details.

Note

If the parameter of interest is the scale parameter, all calculations are performed on the logarithmic scale, though most results are reported on the original scale.

In rare occasions, cond.rsm dumps because of non-convergence of the function rsm which is used to refit the model for a fixed value of the parameter of interest. This happens for instance if this value is too extreme. The arguments from and to may then be used to limit the default range of MLE +/- 3.5 * S.E. A further possibility is to fine-tuning the constants (number of iterations, convergence threshold) that control the rsm fit through the control argument.

cond.rsm may also dump if the estimate of the parameter of interest is large (tipically > 400) in absolute value. This may be avoided by reparametrizing the model.

References

Barndorff-Nielsen, O. E. (1991) Modified signed log likelihood ratio. *Biometrika*, **78**, 557–564.

Brazzale, A. R. (1999) Approximate conditional inference for logistic and loglinear models. *J. Comput. Graph. Statist.*, **8**, 653–661.

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

DiCiccio, T. J., Field, C. A. and Fraser, D. A. S. (1990) Approximations of marginal tail probabilities and inference for scalar parameters. *Biometrika*, **77**, 77–95.

DiCiccio, T. J. and Field, C. A. (1991) An accurate method for approximate conditional and Bayesian inference about linear regression models from censored data. *Biometrika*, **78**, 903–910.

See Also

`marg.object`, `summary.marg`, `plot.marg`, `rsm`

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
##
## quadratic model fitted to the sea level, includes 18.62-year
## astronomical tidal cycle and 11-year sunspot cycle
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
names(coef(venice.rsm))
## "(Intercept)"  "Year"  "I(Year^2)"  "c11"  "s11"  "c19"  "s19"
##
## variable of interest: quadratic term
venice.marg <- cond(venice.rsm, I(Year^2))
##
detach()

## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
##
## parameter of interest: scale parameter
houses.marg <- cond(houses.rsm, scale)
```

---

darwin *Darwin's Data on Growth Rates of Plants*

Description

The `darwin` data frame has 15 rows and 3 columns.

Charles Darwin conducted an experiment to examine the superiority of cross-fertilized plants over self-fertilized plants. 15 pairs of plants were used. Each pair consisted of one cross-fertilized plant and one self-fertilized plant which germinated at the same time and grew in the same pot. The heights of the plants were measured at a fixed time after planting. Four different pots were used.

Usage

```
data(darwin)
```

Format

This data frame contains the following columns:

`cross` the heights of the cross-fertilized plants (in inches);

`self` the heights of the self-fertilized plants (in inches);

`pot` a factor variable for the pot number.

Source

The data were obtained from

Andrews, D. F. and Herzberg, A. M. (1985) *Data: A Collection of Problems From Many Fields for the Student and Research Worker* (Chapter 2). New York: Springer-Verlag.

References

Darwin, C. (1878) *The Effects of Cross and Self Fertilisation in the Vegetable Kingdom* (2nd ed.). London: John Murray.

Examples

```
data(darwin)
plot(cross - self ~ pot, data = darwin)
```

---

family.rsm *Use family() on a "rsm" object*

Description

This is a method for the function `family()` for objects from which a `familyRsm` object can be extracted. Typically a fitted `rsm` model object. See `family` for the general behaviour of this function.

Usage

```
## S3 method for class 'rsm'
family(object, ...)
```

Arguments

`object`	any object from which a `familyRsm` object can be extracted.
`...`	absorbs any additional argument.

See Also

`familyRsm.object`, `family`

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
family(venice.rsm)
detach()

## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
family(houses.rsm)
```

---

familyRsm.object *Family Object for Regression-Scale Models*

Description

Class of objects that characterize the error distribution of a regression-scale model.

Generation

This class of objects is returned by a call to a `family.rsm` generator function. See `rsm.families` for the distributions which are supported by the `marg` package. The object includes a list of functions and expressions that characterize the error distribution of a regression-scale model. These are used by the IRLS algorithm implemented in the `rsm` fitting routine. New families can be added to the ones already supported. See the demonstration file '`margdemo.R`' that ships with the package. There is a `print` method for `familyRsm` objects which produces a simple summary without any detail; use `unclass(familyRsm.object)` to see the contents.

Structure

The following components, with the corresponding functionality, are required for a `familyRsm` object:

`family` a character vector giving the family name.

`g0` a function that yields minus the log density of the error distribution in the regression-scale model.

`g1` a function that yields the first derivative of minus the log density.

`g2` a function that yields the second derivative of minus the log density.

`df` argument with `NULL` value; must be included to guarantee compatibility with the existing code.

`k` argument with `NULL` value; must be included to guarantee compatibility with the existing code.

Note

For the sake of compatibility, the `g0`, `g1` and `g2` functions of a user-defined family can only take two arguments: `y` representing an observation and the `...` argument which absorbes any additional arguments.

See Also

`rsm.families`, `family.rsm`, `rsm`

---

houses *House Price Data*

Description

The `houses` data frame has 26 rows and 5 columns.

Ms. Terry Tasch of Long-Kogan Realty, Chicago, provides data on the selling prices of houses and on different variables describing their status. This data frame contains the prices and a subset of the covariates.

Usage

```
data(houses)
```

Format

This data frame contains the following columns:

`price` selling price (in thousands of dollars);

`bdroom` number of bedrooms;

`floor` floor space (in square feet);

`rooms` total number of rooms;

`front` front footage of lot (in feet).

Source

The data were obtained from

Sen, A. and Srivastava, M. (1990) *Regression Analysis: Theory, Methods and Applications* (Exhibit 2.2, page 32). New York: Springer-Verlag.

Examples

```
data(houses)
summary(houses)
pairs(houses)
```

Huber *Huber's Least Favourable Distribution*

Description

Density, cumulative distribution, quantiles and random number generator for Huber's least favourable distribution.

Usage

```
dHuber(x, k = 1.345)
pHuber(q, k = 1.345)
qHuber(p, k = 1.345)
rHuber(n, k = 1.345)
```

Arguments

x	vector of quantiles. Missing values (NAs) are allowed.
q	vector of quantiles. Missing values (NAs) are allowed.
p	vector of probabilities. Missing values (NAs) are allowed.
n	sample size. If length(n) is larger than 1, then length(n) random values are returned.
k	tuning constant. Values should preferably lie between 1 and 1.5. The default is 1.345, which gives 95% efficiency at the Normal.

Details

Inversion of the cumulative distribution function is used to generate deviates from Huber's least favourable distribution.

Value

Density (dHuber), probability (pHuber), quantile (qHuber), or random sample (rHuber) for Huber's least favourable distribution with tuning constant k. If values are missing, NAs will be returned.

Side Effects

The function rHuber causes creation of the dataset .Random.seed if it does not already exist; otherwise its value is updated.

Background

Huber's least favourable distribution is a compound distribution with gaussian behaviour in the interval (-k,k) and double exponential tails. It is strongly related to Huber's M-estimator, which represents the maximum likelihood estimator of the location parameter.

References

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J. and Stahel, W. A. (1986) *Robust Statistics: The Approach Based on Influence Functions*. New York: Wiley.

Examples

```
pHuber(0.5)
## 0.680374
pHuber(0.5, k = 1.5)
## 0.6842623
```

`logLik.rsm` *Compute the Log Likelihood for Regression-Scale Models*

Description

Computes the log likelihood for regression-scale models.

Usage

```
## S3 method for class 'rsm'
logLik(object, ...)
```

Arguments

`object`	an object inheriting from class `rsm` representing a fitted regression-scale model.
`...`	absorbs any additional argument.

Details

This is a method for the function `logLik()` for objects inheriting from class `rsm`.

Value

Returns an object class `logLik` which is a number with attributes, `attr(r, "df")` (degrees of freedom) giving the number of parameters (regression coefficients plus scale parameter, if not fixed) in the model.

Note

The default `print` method for `logLik` objects is used.

See Also

`rsm.object`, `logLik`

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
##
logLik(venice.rsm)
detach()
```

`marg.object` *Approximate Marginal Inference Object*

Description

Class of objects returned when performing approximate conditional inference for regression-scale models.

Arguments

Objects of class `marg` are implemented as a list. The following components are included:

`workspace`	a list whose elements are the spline interpolations of several first order and higher order statistics. They are used to implement the following likelihood quantities: - the profile and modified profile/approximate marginal log likelihoods; - the Wald pivots from the unconditional and conditional/approximate marginal MLEs; - the profile and modified/approximate marginal likelihood roots; - the conditional/approximate marginal Lugannani-Rice tail area approximation; - the correction term used in the higher order statistics; - the conditional/marginal information and nuisance parameter aspects. Method functions work mainly on this part of the object. In order to avoid errors in the calculation of confidence intervals and tail probabilities, this part of the object should not be modified.
`coefficients`	a 2×2 matrix containing the unconditional and approximate conditional/marginal MLEs and their standard errors.
`call`	the function call that created the `marg` object.
`formula`	the model formula.
`family`	the name of the error distribution.
`offset`	the covariate occurring in the model formula whose coefficient represents the parameter of interest or `scale` if the parameter of interest is the scale parameter.

`diagnostics`	diagnostics related to the decomposition of the higher order adjustments into an information and a nuisance parameters term.
`n.approx`	the number of output points for which the statistics have been calculated exactly.
`omitted.val`	the range of values omitted in the spline interpolation of some of the higher order statistics considered. The aim is to avoid numerical instabilities around the maximum likelihood estimate.
`is.scalar`	a logical value indicating whether there are any nuisance parameters. If `FALSE` there are none.

Main references for the methods considered are the papers by *Barndorff-Nielsen (1991)*, *DiCiccio, Field and Fraser (1990)* and *DiCiccio and Field (1991)*. The theory and statistics used are summarized in *Brazzale (2000, Chapters 2 and 3)*. More details of the implementation and of the methods considered are given in *Brazzale (1999; 2000, Section 6.3.1)*.

Generation

This class of objects is returned from calls to the function `cond.rsm`.

Methods

The class `marg` has methods for `summary`, `plot`, `print`, `coef` and `family`, among others.

Note

If the parameter of interest is the scale parameter, all calculations are performed on the logarithmic scale, though most results are reported on the original scale.

References

Barndorff-Nielsen, O. E. (1991) Modified signed log likelihood ratio. *Biometrika*, **78**, 557–564.

Brazzale, A. R. (1999) Approximate conditional inference for logistic and loglinear models. *J. Comput. Graph. Statist.*, **8**, 653–661.

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

DiCiccio, T. J., Field, C. A. and Fraser, D. A. S. (1990) Approximations of marginal tail probabilities and inference for scalar parameters. *Biometrika*, **77**, 77–95.

DiCiccio, T. J. and Field, C. A. (1991) An accurate method for approximate conditional and Bayesian inference about linear regression models from censored data. *Biometrika*, **78**, 903–910.

See Also

`cond.rsm`, `summary.marg`, `plot.marg`

nuclear *Nuclear Power Station Data*

Description

This data frame contains data on the construction of 32 light water reactors constructed in the USA.

Usage

```
data(nuclear)
```

Format

A data frame with 32 observations on the following 11 variables.

`cost` cost of construction (in billions of dollars adjusted to a 1976 base)

`date` date at which the construction permit was issued

`T1` time between application for and issue of permit

`T2` time between issue of operating license and construction permit

`cap` power plant capacity (in MWe)

`PR` 1 if light water reactor already present on site

`NE` 1 if constructed in north-east region of USA

`CT` 1 if cooling tower used

`BW` 1 if nuclear steam supply system manufactured by Babcock-Wilcox

`N` cumulative number of power plants constructed by each architect-engineer

`PT` 1 if partial turnkey plant

Source

The data were obtained from

Cox, D.R. and Snell, E.J. (1981). *Applied Statistics* (page 81, Example G). Chapman and Hall, London.

Examples

```
data(nuclear)
```

plot.marg *Generate Plots for an Approximate Marginal Inference Object*

Description

Creates a set of plots for an object of class marg.

Usage

```
## S3 method for class 'marg'
plot(x = stop("nothing to plot"), from = x.axis[1], to = x.axis[n],
     which = NULL, alpha = 0.05, add.leg = TRUE, loc.leg = FALSE,
     add.labs = TRUE, cex = 0.7, cex.lab = 1, cex.axis = 1,
     cex.main = 1, lwd1 = 1, lwd2 = 2, lty1 = "solid",
     lty2 = "dashed", col1 = "black", col2 = "blue", tck = 0.02,
     las = 1, adj = 0.5, lab = c(15, 15, 5), ...)
```

Arguments

x	a marg object. This is assumed to be the result returned by the cond.rsm function.
from	the starting value for the x-axis range. The default value has been set by cond.rsm.
to	the ending value for the x-axis range. The default value has been set by cond.rsm.
which	which plot to print. Admissible values are 2 to 7 corresponding to the choices in the menu below.
alpha	the level used to read off confidence intervals; the default is 5%.
add.leg	if TRUE, a legend is added to each plot; default is TRUE.
loc.leg	if TRUE, the position of the legend can be located by hand; default is FALSE.
add.labs	if TRUE, labels are added; default is TRUE.
cex, cex.lab, cex.axis, cex.main	the character expansions relative to the standard size of the device to be used for printing text, labels, axes and main title. See par for details.
lwd1, lwd2	the line widths used to compare different curves in the same plot; default is lwd2 = 2 for higher order solutions and lwd1 = 1 for first order solutions.
lty1, lty2	line type used to compare different curves in the same plot; default is lty2 = "dashed" for the Wald statistic and lty1 = "solid" for the remaining first and higher order statistics.
col1, col2	colors used to compare different curves in the same plot; default is col2 = "blue" for higher order solutions, and col1 = "black" for the remaining first order statistics.
tck, las, adj, lab	further graphical parameters. See par for details.
...	optional graphical parameters; see par for details.

Details

Several plots are produced for an object of class `marg`. A menu lists all the plots that can be produced. They may be one or all of the following ones:

```
 Make a plot selection (or 0 to exit)

1: All
2: Profile and modified profile log likelihoods
3: Profile and modified profile likelihood ratios
4: Profile and modified likelihood root
5: Lugannani-Rice approximation
6: Confidence intervals
7: Diagnostics based on INF/NP decomposition

Selection:
```

If no nuisance parameters are presented, a subset of the above pictures is produced. A message is printed if this is the case. More details and examples are given in *Brazzale (2000, Sections 6.5 and 5.3.2)*.

This function is a method for the generic function `plot()` for class `marg`. It can be invoked by calling `plot` or directly `plot.marg` for an object of the appropriate class.

Value

A plot is created on the current graphics device.

Side Effects

The current device is cleared. When `add.leg` is `TRUE`, a legend is added to each plot. Furthermore, if `loc.leg` is `TRUE`, the location of the legend can be set by the user. All screens are closed, but not cleared, on termination of the function.

Note

If the parameter of interest is the scale parameter, all calculations are performed on the log scale, though most results are reported on the original scale.

Four diagnostic plots are provided. The two panels on the right trace the information and nuisance correction terms, INF and NP, against the likelihood root statistic. These are generally smooth functions and used to approximate the information and nuisance parameter aspects as a function of the parameter of interest, as shown in the two panels on the left. This procedure has the advantage of largely eliminating the numerical instabilities that affect the statistics around the MLE. All four pictures are intended to give an idea of the order of magnitude of the two correction terms while trying to deal with the numerical problems that likely occur for these kinds of data.

More details can be found in *Brazzale (2000, Appendix B.2)*.

References

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

See Also

cond.rsm, marg.object, summary.marg

Examples

```
# Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
#
# quadratic model fitted to the sea level, includes 18.62-year
# astronomical tidal cycle and 11-year sunspot cycle
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
venice.marg <- cond(venice.rsm, I(Year^2))
plot(venice.marg, which = 4)
##
detach()
```

---

print.summaryMarg *Use print() on a "summaryMarg" object*

Description

This is a method for the function print() for objects of class summaryMarg. See print and print.default for the general behaviour of this function and for the interpretation of digits.

Usage

```
## S3 method for class 'summaryMarg'
print(x, all = x$all, Coef = x$cf, int = x$int, test = x$hyp,
     digits = if(!is.null(x$digits)) x$digits else max(3, getOption("digits")-3),
       ...)
## S3 method for class 'summaryMarg'
print(x, all, Coef, int, test, digits, ...)
```

Arguments

x	a summaryMarg object. This is assumed to be the result returned by the summary.marg function.

all	if TRUE all the information stored in the summaryMarg object is printed, else only a subset of it. The default is FALSE.
Coef	if TRUE all parameter estimates are printed. The default is TRUE.
int	if TRUE confidence intervals are printed. The default is TRUE.
test	if TRUE test statistics and tail probabilities are printed. The default is FALSE.
digits	the number of significant digits to be printed. The default depends on the value of digits set by options.
...	additional arguments.

Details

Changing the default values of all, Coef, int and test allows only a subset of the information in the summaryMarg object to be printed. With all = FALSE, one-sided confidence intervals and the Lugannani-Rice tail area approximation are omitted. See summary.marg for more details.

Note

If the parameter of interest is the scale parameter, all calculations are performed on the log scale, though most results are reported on the original scale.

The amount of information printed may vary depending on whether there are any nuisance parameters. A message is printed if there are none.

See Also

summary.marg, marg.object

Examples

```
## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
houses.cond <- cond(houses.rsm, front)
print(summary(houses.cond), digits = 4)
print(summary(houses.cond), Coef = FALSE)
```

---

residuals.rsm *Compute Residuals for Regression-Scale Models*

Description

Computes one of the six types of residuals available for regression-scale models.

Usage

```
## S3 method for class 'rsm'
residuals(object, type = c("deviance", "pearson",
                           "response", "r.star", "prob", "deletion"),
          weighting = "observed", ...)
```

Arguments

`object`	an object inheriting from class `rsm` representing a fitted regression-scale model.
`type`	character string; defines the type of residuals, with choices `"deviance"`, `"pearson"`, `"response"`, `"r.star"`, `"prob"` or `"deletion"`; the first is the default.
`weighting`	character string; defines the weight matrix that should be used in the calculation of the residuals and diagnostics. Possible choices are `"observed"`, `"score"`, `"deviance"` and `"max"`; see *Jorgensen (1984)* for their definition. The default is `"observed"`.
`...`	absorbs any additional argument.

Details

This is a method for the function `residuals()` for objects inheriting from class `rsm`. As several types of residuals are available for `rsm` objects, there is an additional optional argument `type`. The `"deviance"`, `"pearson"`, `"r.star"`, `"prob"` and `"deletion"` residuals are derived from the final IRLS fit. The `"response"` residuals are standardized residuals on the scale of the response, the `"prob"` residuals are on the $U(0,1)$ scale, whereas the remaining ones follow approximately the standard normal distribution.

The default weighting scheme used is `"observed"`. The weights used are the values stored in the `q2` component of the `rsm` object. Some of the IRLS weights returned by `rsm` may be negative if the error distribution is Student's t or user-defined. In order to avoid missing values in the residuals, the default weighting scheme used is then `"score"` unless otherwise specified. The `"score"` weights are also used by default if Huber's least favourable error distribution is used.

More details, in particular of the use of these residuals, are given in *Brazzale (2000, Section 6.3.1).*

Value

A numeric vector of residuals. See *Davison and Snell (1991)* for detailed definitions of each type of residual.

Note

The `summary` method for `rsm` objects produces response residuals. The `residuals` component of a `rsm` object contains the response residuals.

References

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

Davison, A. C. and Snell, E. J. (1991) Residuals and diagnostics. In *Statistical Theory and Modelling: In Honour of Sir David Cox* (eds. D.V. Hinkley, N. Reid, and E.J. Snell), 83–106. London: Chapman & Hall.

Davison, A. C. and Tsai, C.-L. (1992) Regression model diagnostics. *Int. Stat. Rev.*, **60**, 337–353.

Jorgensen, B. (1984). The delta algorithm and GLIM. *Int. Stat. Rev.*, **52**, 283–300.

See Also

rsm.object, residuals

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
##
residuals(venice.rsm)
## deviance residuals with observed weights
residuals(venice.rsm, type = "r.star", weighting = "score")
## r* residuals with score weights
detach()
```

---

rsm *Fit a Regression-Scale Model*

Description

Produces an object of class rsm which is a regression-scale model fit of the data.

Usage

```
rsm(formula = formula(data), family = gaussian,
    data = sys.frame(sys.parent()), dispersion = NULL,
    weights = NULL, subset = NULL, na.action = na.fail,
    offset = NULL, method = "rsm.surv",
    control = glm.control(maxit=100, trace=FALSE),
    model = FALSE, x = FALSE, y = TRUE, contrasts = NULL, ...)
```

Arguments

formula	a formula expression as for other linear regression models, of the form response ~ predictors where the predictors are separated by suitable operators. See the documentation of lm and formula for details.
family	a family.rsm object, i.e. a list of functions and expressions characterizing the error distribution. Families supported are gaussian, student (Student's t), extreme (Gumbel or extreme value), logistic, logWeibull, logExponential, logRayleigh and Huber (Huber's least favourable). These represent calls to the corresponding generator functions. The calls to gaussian, extreme, logistic, logWeibull, logExponential and logRayleigh can be given without parentheses. The functions student and Huber may take as argument respectively the

degrees of freedom (df) and the tuning constant (k). Users can construct their own families, as long as they have components compatible with those given in rsm.distributions. The demonstration file 'margdemo.R' that ships with the package shows how to create a new generator function. The default is gaussian.

data — an optional data frame in which to interpret the variables occurring in the model formula, or in the subset and the weights arguments. If this is missing, then the variables in the formula should be on the search list.

dispersion — if NULL, the scale parameter is taken to be unknown. If known, the numerical value can be passed. The default is NULL. Huber's least favourable distribution represents a special case. If dispersion is NULL, the maximum likelihood estimate is computed, while if TRUE the MAD estimate is calculated and the scale parameter fixed to this value in subsequent computations.

weights — the optional weights for the fitting criterion. If supplied, the response variable and the covariates are multiplied by the weights in the IRLS algorithm. The length of the weights argument 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 subset argument.

subset — 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.

na.action — a function to filter missing data. This is applied to the model frame after any subset argument has been used. The default (with na.fail) is to create an error if any missing value is found. A possible alternative is na.omit, which deletes observations that contain one or more missing values.

offset — this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. An offset term can be included in the formula instead or as well, and if both are specified their sum is used. Defaults to NULL

method — the fitting method to be used; the default is rsm.fit. The method model.frame simply returns the model frame.

control — a list of iteration and algorithmic constants. See glm.control for their names and default values.

model — if TRUE, the model frame is returned; default is FALSE.

x — if TRUE, the model matrix is returned; default is FALSE.

y — if TRUE, the response variable is returned; default is TRUE.

contrasts — 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.

... — absorbs any additional argument.

Details

The model is fitted using *Iteratively Reweighted Least Squares*, IRLS for short (*Green, 1984*, *Jorgensen, 1984*). The working response and iterative weights are computed using the functions contained in the `family.rsm` object.

The two workhorses of `rsm` are `rsm.fit` and `rsm.surv`, which expect an `X` and `Y` argument rather then a formula. The first function is used for the families `student` with `df` < 3 and `Huber`; the second one, based on the `survreg.fit` routine for fitting parametric survival models, is used in case of `extreme`, `logistic`, `logWeibull`, `logExponential`, `logRayleigh` and `student` (with `df` > 2) error distributions. In the presence of a user-defined error distribution the `rsm.fit` routine is used. The `rsm.null` function is invoked to fit an empty (null) model.

The details are given in *Brazzale (2000, Section 6.3.1)*.

Value

an object of class `rsm` is returned which inherits from `glm` and `lm`. See `rsm.object` for details.

The output can be examined by `print`, `summary`, `rsm.diag.plots` and `anova`. Components can be extracted using `fitted`, `residuals`, `formula` and `family`. It can be modified using `update`. It has most of the components of a `glm` object, with a few more. Use `rsm.object` for further details.

Note

In case of `extreme`, `logistic`, `logWeibull`, `logExponential`, `logRayleigh` and `student` (with `df` > 2) error distributions, both methods, `rsm.fit` (default choice) and `rsm.surv`, can be used to fit the model. There are, however, examples where one of the two algorithms (most likely the one invoked by `rsm.surv`) breaks down. If this is the case, try and refit the model with the alternative choice.

The message `"negative iterative weights returned!"` is returned if some of the iterative weights (`q2` component of the fitted `rsm` object) are negative. These would be used by default by the `rsm.diag` routine for the definition of residuals and regression diagnostics. In order to avoid missing values (`NA`s), the default weighting scheme `"observed"` automatically switches to `"score"` unless otherwise specified.

References

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

Green, P. J. (1984) Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives (with Discussion). *J. R. Statist. Soc.* B, **46**, 149–192.

Jorgensen, B. (1984) The delta algorithm and GLIM. *Int. Stat. Rev.*, **52**, 283–300.

See Also

`rsm.object`, `rsm.fit`, `rsm.surv`, `rsm.null`, `rsm.families`

Examples

```
## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
## model fit including all covariates
houses.rsm <- rsm(price ~ ., family = student(5), data = houses,
                  method = "rsm.fit", control = glm.control(trace = TRUE))
## prints information about the iterative procedure at each iteration
update(houses.rsm, ~ . - bdroom + offset(7 * bdroom))
## "bdroom" is included as offset variable with fixed (= 7) coefficient

## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
venice.2.rsm <- rsm(sea ~ Year + I(Year^2), family = extreme)
## quadratic model fitted to sea level data
venice.1.rsm <- update(venice.2.rsm, ~. - I(Year^2))
## linear model fit
##
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
## includes 18.62-year astronomical tidal cycle and 11-year sunspot cycle
venice.11.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11, family = extreme)
venice.19.rsm <- rsm(sea ~ Year + I(Year^2) + c19 + s19, family = extreme)
## includes either astronomical cycle
##
## comparison of linear, quadratic and periodic (11-year, 19-year) models
plot(year, sea, ylab = "sea level")
lines(year, fitted(venice.1.rsm))
lines(year, fitted(venice.2.rsm), col="red")
lines(year, fitted(venice.11.rsm), col="blue")
lines(year, fitted(venice.19.rsm), col="green")
##
detach()

## Darwin's Data on Growth Rates of Plants
data(darwin)
darwin.rsm <- rsm(cross - self ~ pot - 1, family  =  student(3),
                  data = darwin)
## Maximum likelihood estimates
darwin.rsm <- rsm(cross - self ~ pot - 1, family = Huber, data = darwin)
## M-estimates
```

`rsm.diag` *Diagnostics for Regression-Scale Models*

Description

Calculates different types of residuals, Cook's distance and the leverages for a regression-scale model.

Usage

```
rsm.diag(rsmfit, weighting = "observed")
```

Arguments

`rsmfit`	an `rsm` object, i.e. the result of a call to `rsm`.
`weighting`	character string; defines the weight matrix that should be used in the calculation of the residuals and diagnostics. Possible choices are `"observed"`, `"score"`, `"deviance"` and `"max"`; see *Jorgensen (1984)* for their definition. The default is `"observed"`.

Details

If the weighting scheme is `"observed"`, the weights used are the values stored in the `q2` component of the `rsm` object `rsmfit`. Otherwise, they are calculated by `rsm.diag`. Some of the IRLS weights returned by `rsm` may be negative if the error distribution is Student's t or user-defined. In order to avoid missing values in the residuals and regression diagnostics, the default weighting scheme used in `rsm.diag` switches automatically from `"observed"` to `"score"` unless otherwise specified. The `"score"` weights are also used by default if Huber's least favourable error distribution is used.

There are three types of residuals. The response residuals are taken on the response scale, whereas the probability transform residuals are on the $U(0,1)$ scale. The remaining ones follow approximately the standard normal distribution.

More details and in particular the definitions of the above residuals and diagnostics can be found in *Brazzale (2000, Section 6.3.1).*

Value

Returns a list with the following components:

`resid`	the response residuals on the response scale.
`rd`	the standardized deviance residuals from the IRLS fit.
`rp`	the standardized Pearson residuals from the IRLS fit.
`rg`	the deletion residuals from the IRLS fit.
`rs`	the r^* residuals from the IRLS fit.
`rcs`	the probability transform residuals from the IRLS fit.
`cook`	Cook's distance.
`h`	the leverages of the observations.
`dispersion`	the value of the scale parameter.

Acknowledgments

This function is based on A.J. Canty's function `glm.diag` contained in the package **boot**.

Note

Huber's least favourable distribution represents a special case. The regression diagnostics are only meaningful if the errors *truly* follow a Huber-type distribution. This no longer holds if the option `family = Huber` in `rsm` is used to obtain the M-estimates of the parameters in place or the maximum likelihood estimates.

References

Brazzale, A. R. (2000) *Practical Small-Sample Parametric Inference*. Ph.D. Thesis N. 2230, Department of Mathematics, Swiss Federal Institute of Technology Lausanne.

Jorgensen, B. (1984) The delta algorithm and GLIM. *Int. Stat. Rev.*, **52**, 283–300.

Davison, A. C. and Snell, E. J. (1991) Residuals and diagnostics. In *Statistical Theory and Modelling: In Honour of Sir David Cox* (eds. D. V. Hinkley, N. Reid, and E. J. Snell), 83–106. London: Chapman & Hall.

Davison, A. C. and Tsai, C.-L. (1992) Regression model diagnostics. *Int. Stat. Rev.*, **60**, 337–353.

See Also

`rsm.diag.plots`, `rsm.object`, `summary.rsm`

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
venice.diag <- rsm.diag(venice.rsm)
## observed weights
detach()

## Darwin's Data on Growth Rates of Plants
data(darwin)
darwin.rsm <- rsm(cross-self ~ pot - 1, family = Huber, data = darwin)
darwin.diag <- rsm.diag(darwin.rsm)
## score weights
```

`rsm.diag.plots` *Diagnostic Plots for Regression-Scale Models*

Description

Generates diagnostic plots for a regression-scale model using different types of residuals, Cook's distance and the leverages.

Usage

```
rsm.diag.plots(rsmfit, rsmdiag = NULL, weighting = NULL,
               which = NULL, subset = NULL, iden = FALSE,
               labels = NULL, ret = FALSE, ...)
## S3 method for class 'rsm'
plot(x, ...)
```

Arguments

rsmfit, x	a rsm object, i.e. the result of a call to rsm.
rsmdiag	the object returned by a call to rsm.diag containing the regression diagnostics for the regression-scale model defined by rsmfit. If not supplied, this object is created by rsm.diag.plots and returned upon request (if ret = TRUE).
weighting	character string; defines the weight matrix that should be used in the calculation of the residuals and diagnostics. Possible choices are "observed", "score", "deviance" and "max"; see *Jorgensen (1984)* for their definition. Will only be used if the rsmdiag argument is missing.
which	which plot to print. Admissible values are 2 to 7 corresponding to the choices in the menu below.
subset	subset of data used in the original rsm fit: should be the same than the subset option used in the call to rsm which generated rsmfit. Needed only if the subset option was used in the call to rsm.
iden	logical argument. If TRUE, the user will be prompted after the plots are drawn. A positive integer will select a plot and invoke identify() on that plot. After exiting identify(), the user is again prompted, this loop continuing until the user responds to the prompt with 0. If iden is FALSE (default) the user cannot interact with the plots.
labels	a vector of labels for use with identify() if iden is TRUE. If it is not supplied, then the labels are derived from rsmfit.
ret	logical argument indicating if rsmdiag should be returned; the default is FALSE.
...	additional arguments such as graphical parameters.

Details

The diagnostics required for the plots are calculated by rsm.diag. These are then used to produce the plots on the current graphics device.

A menu lists all the plots that can be produced. They may be one or all of the following:

```
 Make a plot selection (or 0 to exit)

1: All
2: Response residuals against fitted values
3: Deviance residuals against fitted values
4: QQ-plot of deviance residuals
5: Normal QQ-plot of r* residuals
```

```
6: Cook statistic against h/(1-h)
7: Cook statistic against observation number

Selection:
```

In the normal scores plots, the dotted line represents the expected line if the residuals are normally distributed, i.e. it is the line with intercept 0 and slope 1.

In general, when plotting Cook's distance against the standardized leverages, there will be two dotted lines on the plot. The horizontal line is at $8/(n-2p)$, where n is the number of observations and p is the number of estimated parameters. Points above this line may be points with high influence on the model. The vertical line is at $2p/(n-2p)$ and points to the right of this line have high leverage compared to the variance of the raw residual at that point. If all points are below the horizontal line or to the left of the vertical line then the line is not shown.

Use of `iden = TRUE` is encouraged for proper exploration of these plots as a guide to how well the model fits the data and whether certain observations have an unduly large effect on parameter estimates.

Value

If `ret` is `TRUE` then the value of `rsmdiag` is returned, otherwise there is no returned value.

Side Effects

The current device is cleared. If `iden = TRUE`, interactive identification of points is enabled. All screens are closed, but not cleared, on termination of the function.

Acknowledgments

This function is based on A. J. Canty's function `glm.diag.plots` contained in the package **boot**.

References

Davison, A. C. and Snell, E. J. (1991) Residuals and diagnostics. In *Statistical Theory and Modelling: In Honour of Sir David Cox* (eds. D. V. Hinkley, N. Reid, and E. J. Snell), 83–106. London: Chapman & Hall, London.

Davison, A. C. and Tsai, C.-L. (1992) Regression model diagnostics. *Int. Stat. Rev.*, **60**, 337–353.

Jorgensen, B. (1984) The Delta Algorithm and GLIM. *Int. Stat. Rev.*, **52**, 283–300.

See Also

`rsm.diag`, `rsm.object`, `identify`

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
```

```
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
## Not run:
rsm.diag.plots(venice.rsm, which = 3)

## End(Not run)
## or
## Not run:
plot(venice.rsm)

## End(Not run)
## menu-driven
##
rsm.diag.plots(venice.rsm, which = 5, las = 1)
## normal QQ-plot of r* residuals
## Not run:
rsm.diag.plots(venice.rsm, which = 7, iden = T, labels = paste(1931:1981))

## End(Not run)
## year 1932 highly influential
detach()
```

`rsm.families` *Generate a RSM Family Object*

Description

Generates a `familyRsm` object containing a list of functions and expressions used by `rsm`.

Usage

```
extreme()
Huber(k = 1.345)
logistic()
logWeibull()
student(df = stop("Argument \"df\" is missing, with no default"))
```

Arguments

`k`	the tuning constant in Huber's least favourable distribution.
`df`	the degrees of freedom in Student's t distribution.

Details

Each of the names are associated with a member of the class of error distributions for regression-scale models. Users can construct their own families, as long as they have components compatible with those given in `rsm.distributions`. The demonstration file '`margdemo.R`' that accompanies

the package shows how to create a new generator function. When passed as an argument to `rsm` with the default setting, the empty parentheses `()` can be omitted. There is a `print` method for the class `familyRsm`.

Value

A `familyRsm` object, which is a list of functions and expressions used by `rsm` in the iteratively reweighed least-squares algorithm. See `familyRsm.object` for details.

See Also

`familyRsm.object`, `family.rsm`, `rsm`, `Huber`

Examples

```
student(df = 3) ## generates Student's t error distribution with 3 d.f.
## Not run:
rsm(formula = value, data = value, family = extreme)

## End(Not run)
```

---

rsm.fit *Fit a Regression-Scale Model Without Computing the Model Matrix*

Description

Fits a `rsm` model without computing the model matrix of the response vector.

Usage

```
rsm.fit(X, Y, offset, family, dispersion, score.dispersion, maxit, epsilon,
        trace, ...)
```

Arguments

`X`	the model matrix (design matrix).
`Y`	the response vector.
`dispersion`	if `NULL`, the MLE of the scale parameter is returned, otherwise the scale parameter is fixed to the numerical value passed through the argument. If Huber's least favourable distribution is used and `dispersion` is `TRUE`, the MAD is computed and the scale parameter fixed to this value in subsequent calculations.
`score.dispersion`	must default to `NULL`.
`offset`	optional offset added to the linear predictor.

family	a family.rsm object, i.e. a list of functions and expressions characterizing the error distribution. Families supported are gaussian, student (Student's t), extreme (Gumbel or extreme value), logistic, logWeibull, logExponential, logRayleigh and Huber (Huber's least favourable). Users can construct their own families, as long as they have components compatible with those given in rsm.distributions. The demonstration file 'margdemo.R' that ships with the package shows how to create a new generator function.
maxit	maximum number of iterations allowed.
epsilon	convergence threshold.
trace	if TRUE, iterations details are printed during execution.
...	not used, but do absorb any redundant argument.

Details

The rsm.fit function is called internally by the rsm routine to do the actual model fitting. Although it is not intended to be used directly by the user, it may be useful when the same data frame is used over and over again. It might save computational time, since the model matrix is not created. No formula needs to be specified as an argument. As no weights argument is available, the response Y and the model matrix X must already include the weights if weighting is desired.

Value

an object which is a subset of a rsm object.

Note

The rsm.fit function is the workhorse of the rsm fitting routine for the student (with df ≤ 2), Huber and user-defined error distributions. It receives X and Y data rather than a formula, but still uses the family.rsm object to define the IRLS steps. Users can write their own versions of rsm.fit, and pass the name of their function via the method argument to rsm. Care should be taken to include as many of the arguments as feasible, but definitely the ... argument, which will absorb any additional argument given in the call from rsm.

See Also

rsm, rsm.surv, rsm.null, rsm.object, rsm.families

---

rsm.null — *Fit an Empty Regression-Scale Model*

Description

Fits a rsm model with empty model matrix.

Usage

```
rsm.null(X = NULL, Y, offset, family, dispersion, score.dispersion, maxit,
         epsilon, trace, ...)
```

Arguments

X	defaults to NULL.
Y	the response vector.
dispersion	either NULL or TRUE. If NULL, the MLE of the scale parameter is returned. If Huber's least favourable distribution is used and dispersion is TRUE, the MAD is computed and the scale parameter fixed to this value in subsequent calculations.
score.dispersion	must default to NULL.
offset	optional offset added to the linear predictor.
family	a family.rsm object, i.e. a list of functions and expressions characterizing the error distribution. Families supported are gaussian, student (Student's t), extreme (Gumbel or extreme value), logistic, logWeibull, logExponential, logRayleigh and Huber (Huber's least favourable). Users can construct their own families, as long as they have components compatible with those given in rsm.distributions. The demonstration file 'margdemo.R' that ships with the package shows how to create a new generator function.
maxit	maximum number of iterations allowed.
epsilon	convergence threshold.
trace	if TRUE, iterations details are printed during execution.
...	not used, but do absorb any redundant argument.

Details

The rsm.null function is called internally by the rsm routine to do the actual model fitting in case of an empty model. It is not intended to be used directly by the user. As no weights argument is available, the response Y and the model matrix X must already include the weights if weighting is desired.

Value

an object which is a subset of a rsm object.

See Also

rsm, rsm.surv, rsm.fit, rsm.object, rsm.families

---

rsm.object *Regression-Scale Model Object*

Description

Class of objects returned when fitting a regression-scale model.

Arguments

The following components must be included in a `rsm` object:

`coefficients`	the coefficients of the linear predictor, 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.
`dispersion`	the (estimated or known) value of the scale parameter.
`fixed`	a logical value. If `TRUE`, the scale parameter is fixed.
`residuals`	the response residuals from the fit. If weights were used, they are not taken into account. If you need other kinds of residuals, use the `residuals.rsm` function.
`fitted.values`	the fitted values from the fit. If weights were used, the fitted values are not adjusted for the weights.
`loglik`	the log likelihood from the fit.
`q1`	the value of the first derivative of minus the log density for each observation.
`q2`	the value of the second derivative of minus the log density for each observation.
`rank`	the computed rank (number of linearly independent columns in the model matrix).
`R`	the unscaled observed information matrix.
`score.dispersion`	a list containing the value of the objective function, its gradient and the convergence diagnostic, that result from estimating the scale parameter.
`iter`	the number of IRLS iterations used to compute the estimates.
`weights`	the (optional) weights used for the fit.
`assign`	the list of assignments of coefficients (and effects) to the terms in the model. The names of this list are the names of the terms. The *i*th element of the list is the vector saying which coefficients correspond to the *i*th term. It may be of length 0 if there were no estimable effects for the term.
`df.residuals`	the number of degrees of freedom for residuals.
`family`	the entire `family.rsm` object used.
`user.def`	a logical value. If `TRUE`, the error distribution is user-defined.
`dist`	a character string representing the name of the error distribution.
`formula`	the model formula.
`data`	the data frame in which to interpret the variables occurring in the model formula, or in the `subset` and the `weights` arguments to `rsm`.
`terms`	an object of mode `expression` and class `term` summarizing the formula.
`contrasts`	a list containing sufficient information to construct the contrasts used to fit any factors occurring in the model. The list contains entries that are either matrices or character vectors. When a factor is coded by contrasts, the corresponding contrast matrix is stored in this list. Factors that appear only as dummy variables and variables in the model that are matrices correspond to character vectors in the list. The character vector has the level names for a factor or the column labels for a matrix.

control	a list of iteration and algorithmic constants used in rsm to fit the model.
call	an image of the call that produced the object, but with the arguments all named and with the actual formula included as the formula argument.
y	optionally the response, if y = TRUE in the original rsm call.
x	optionally the model matrix, if x = TRUE in the original rsm call.
model	optionally the model frame, if model = TRUE in the original rsm call.

Generation

This class of objects is returned by the rsm function to represent a fitted regression-scale model. Class rsm inherits from classes glm and lm, since it is fitted by iteratively reweighted least squares. The object returned has all the components of a weighted least squares object.

Methods

Objects of this class have methods for the functions print, summary, anova and fitted among others.

Note

The residuals, fitted values and coefficients should be extracted by the generic functions of the same name, rather than by the $ operator.

See Also

rsm, glm, lm.

---

rsm.surv *Fit a Regression-Scale Model Without Computing the Model Matrix*

Description

Fits a rsm model without computing the model matrix of the response vector.

Usage

```
rsm.surv(X, Y, offset, family, dispersion, score.dispersion, maxit, epsilon,
         trace, ...)
```

Arguments

X	the model matrix (design matrix).
Y	the response vector.
offset	optional offset added to the linear predictor.

family	a family.rsm object, i.e. a list of functions and expressions characterizing the error distribution. Families supported are extreme (Gumbel or extreme value), logWeibull, logExponential, logRayleigh, logistic and student (Student's t) with df > 2.
dispersion	if NULL, the MLE of the scale parameter is returned, otherwise the scale parameter is fixed to the numerical value passed through the argument.
score.dispersion	must default to NULL.
maxit	maximum number of iterations.
epsilon	convergence threshold.
trace	if TRUE, iterations details are printed during execution.
...	not used, but do absorb any redundant argument.

Details

The rsm.surv function is called internally by the rsm routine to do the actual model fitting. Although it is not intended to be used directly by the user, it may be useful when the same data frame is used over and over again. It might save computational time, since the model matrix is not created. No formula needs to be specified as an argument. As no weights argument is available, the response Y and the model matrix X must already include the weights if weighting is desired.

Value

an object, which is a subset of a rsm object.

Note

The rsm.surv function is the default option for rsm for the extreme, logistic, logWeibull, logExponential, logRayleigh and student (with df larger than 2) error distributions. It makes use of the survreg.fit routine to estimate parametric survival models. It receives X and Y data rather than a formula, but still uses the family.rsm object to define the IRLS steps. The rsm.surv routine cannot be used for Huber-type and user-defined error distributions.

See Also

rsm, rsm.fit, rsm.null, rsm.object, rsm.families

---

summary.marg *Summary Method for Objects of Class "marg"*

Description

Returns a summary list for objects of class marg.

Usage

```
## S3 method for class 'marg'
summary(object, alpha = 0.05, test = NULL, all = FALSE,
        coef = TRUE, int = ifelse((is.null(test) || all), TRUE, FALSE),
        digits = NULL, ...)
```

Arguments

object	a marg object. This is assumed to be the result returned by the cond.rsm function.
alpha	a vector of levels for confidence intervals; the default is 5%.
test	a vector of values of the parameter of interest one wants to test for. If NULL no test is performed. The default is NULL.
all	logical value; if TRUE all the information stored in the summaryMarg object is printed, else only a subset of it. The default is FALSE.
coef	logical value; if TRUE the unconditional and approximate conditional/marginal MLEs are printed. The default is TRUE.
int	logical value; if TRUE confidence intervals are printed. The default is TRUE.
digits	the number of significant digits to be printed. The default depends on the value of digits set by options
...	absorbs any additional argument.

Details

This function is a method for the generic function summary() for objects of class marg. It can be invoked by calling summary or directly summary.marg for an object of the appropriate class.

Value

A list is returned with the following components:

coefficients	a 2×2 matrix containing the unconditional and approximate conditional/marginal MLEs and their standard errors.
conf.int	a matrix containing, for each level given in alpha, the upper and lower confidence bounds derived from several first and higher order test statistics. One-sided and two-sided confidence intervals are considered. See marg.object for details on the test statistics used.
signif.tests	a list with two elements. The first (stats) contains, for each value given in test, the values and tail probabilities of several first and higher order test statistics. See marg.object for details on the test statistics. The second element of the list (qTerm) contains for each tested hypothesis the correction term used in the higher order solutions.
call	the function call that created the marg object.
formula	the model formula.
family	the name of the error distribution.

offset	the covariate occurring in the model formula whose coefficient represents the parameter of interest or scale if the parameter of interest is the scale parameter.
alpha	the vector of levels used to compute the confidence intervals.
hypotheses	the values for the parameter of interest that have been tested for.
diagnostics	the information and nuisance parameters aspects; see marg.object for details.
n.approx	the number of output points that have been calculated exactly.
all	logical value; if TRUE, all the information stored in the summaryMarg object is printed.
cf	logical value; if TRUE, the parameter estimates are printed.
int	logical value; if TRUE, confidence intervals are printed.
is.scalar	a logical value indicating whether there are any nuisance parameters. If FALSE there are none.
digits	the number of significant digits to be printed.

Note

If the parameter of interest is the scale parameter, all calculations are performed on the log scale, though most results are reported on the original scale.

The amount of information calculated may vary depending on whether there are any nuisance parameters. A message is printed if there are none.

See Also

summary, marg.object

Examples

```
## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
houses.marg <- cond(houses.rsm, floor)
summary(houses.marg, test = 0, coef = FALSE)
```

---

summary.rsm *Summary Method for Regression-Scale Models*

Description

Returns a summary list for a fitted regression-scale model.

Usage

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

Arguments

object	a fitted rsm object. This is assumed to be the result of some fit that produces an object inheriting from the class rsm, in the sense that the components returned by the rsm function will be available.
correlation	logical argument. If TRUE, the correlation matrix for the coefficients is computed; default is TRUE.
digits	a non-null value specifies the minimum number of significant digits to be printed in values. If NULL, the value of digits set by options is used.
...	absorbs any additional argument.

Details

This function is a method for the generic function summary() for class rsm. It can be invoked by calling summary for an object of the appropriate class, or directly by calling summary.rsm regardless of the class of the object.

Value

A list is returned with the following components:

coefficients	a matrix with four columns, containing the coefficients, their standard errors, the z-values (or Wald statistics) and the associated p-values based on the standard normal approximation to the distribution of the z statistic.
dispersion	the value of the scale parameter used in the computations.
fixed	a logical value. If TRUE, the scale parameter is known.
residuals	the response residuals.
cov.unscaled	the unscaled covariance matrix, i.e, a matrix such that multiplying it by the squared scale parameter, or an estimate thereof, produces an estimated (asymptotic) covariance matrix for the coefficients.
correlation	the computed correlation matrix for the coefficients in the model.
family	the entire family.rsm object used.
loglik	the computed log likelihood.
terms	an object of mode expression and class term summarizing the formula.
df	the number of degrees of freedom for the model and for the residuals.
iter	the number of IRLS iterations used to compute the estimates.
nas	a logical vector indicating which, if any, coefficients are missing.
call	an image of the call that produced the rsm object, but with the arguments all named and with the actual formula.
digits	the value of the digits argument.

See Also

rsm.object, summary, rsm

Examples

```
## House Price Data
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
summary(houses.rsm)
```

---

update.rsm *Update and Re-fit a RSM Model Call*

Description

update.rsm is used to update a rsm model formulae. This typically involves adding or dropping terms, but updates can be more general.

Usage

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

Arguments

object	a model of class rsm to be updated.
formula.	changes to the formula – see update.formula for details.
...	additional arguments to the call, or arguments with changed values. Use name = NULL to remove the argument name.
evaluate	if TRUE evaluate the new call else return the call.

Value

If evaluate = TRUE the fitted object, otherwise the updated call.

Note

Based upon update.default.

See Also

update, update.default, update.formula

Examples

```
data(houses)
houses.rsm <- rsm(price ~ ., family = student(5), data = houses)
## model fit including all covariates
##
houses.rsm <- update(houses.rsm, method = "rsm.fit",
                     control = glm.control(trace = TRUE))
```

```
## prints information about the iterative procedure at each iteration
##
update(houses.rsm, ~ . - bdroom + offset(7 * bdroom))
## "bdroom" is included as offset variable with fixed (= 7) coefficient
```

---

vcov.rsm *Calculate Variance-Covariance Matrix for a Fitted RSM Model*

Description

Returns the variance-covariance matrix of the parameters of a fitted rsm model object.

Usage

```
## S3 method for class 'rsm'
vcov(object, correlation = FALSE, ...)
```

Arguments

object	a fitted model object of class rsm.
correlation	if TRUE the correlation matrix is returned instead of the variance-covariance matrix.
...	absobs any additional argument.

Details

This is a method for function vcov for objects of class rsm.

Value

A matrix of the estimated covariances between the parameter estimates of a fitted regression-scale model, or, if dispersion = TRUE the correlation matrix.

See Also

vcov, rsm.object, rsm, summary.rsm

Examples

```
## Sea Level Data
data(venice)
attach(venice)
Year <- 1:51/51
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.rsm <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
                  family = extreme)
##
```

```
vcov(venice.rsm)
vcov(venice.rsm, corr = TRUE)
##
detach()
```

---

venice *Sea Level Data*

Description

The venice data frame has 51 rows and 2 columns.

Pirazzoli (1982) collected the ten largest values of sea levels in Venice (with a few exceptions) for the years 1887–1981. The venice data frame contains the maxima for the years 1931–1981.

Usage

```
data(venice)
```

Format

This data frame contains the following columns:

year the years;

sea the sea levels (in cm).

Source

The data were obtained from

Smith, R. L. (1986) Extreme value theory based on the r-largest annual events. *Journal of Hydrology* , **86**, 27–43.

References

Pirazzoli, P. (1982) Maree estreme a Venezia (periodo 1872–1981). *Acqua Aria*, **10**, 1023–1039.

Examples

```
data(venice)
attach(venice)
#
plot(sea ~ year, ylab = "sea level")
##
Year <- 1:51/51
venice.l <- rsm(sea ~ Year + I(Year^2), family = extreme)
lines(year, fitted(venice.l))
##
c11 <- cos(2*pi*1:51/11) ; s11 <- sin(2*pi*1:51/11)
c19 <- cos(2*pi*1:51/18.62) ; s19 <- sin(2*pi*1:51/18.62)
venice.p <- rsm(sea ~ Year + I(Year^2) + c11 + s11 + c19 + s19,
```

```
                family = extreme)
lines(year, fitted(venice.p), col = "red")
##
detach()
```

Index