

Package ‘lasso2’

April 17, 2009

Version 1.2-10

Date 2009-01-26

Author Justin Lokhorst, Bill Venables and Berwin Turlach; port to R, tests etc: Martin Maechler
<maechler@stat.math.ethz.ch>

Maintainer Berwin Turlach <statba@nus.edu.sg>

Title L1 constrained estimation aka ‘lasso’

Description Routines and documentation for solving regression problems while imposing an L1 constraint on the estimates, based on the algorithm of Osborne et al. (1998)

Depends R (>= 2.5.0)

License GPL (>= 2)

URL <http://www.maths.uwa.edu.au/~berwin/software/lasso.html>

Repository CRAN

Date/Publication 2009-01-26 09:22:43

R topics documented:

aux	2
aux.l1celist	3
coef.l1ce	3
coef.l1celist	4
deviance.g11ce	4
deviance.l1ce	5
Extract.l1celist	5
fitted.l1ce	6
gcv	6
gcv.l1ce	7
g11ce	8
g11ce.object	10
Iowa	11

is.formula	12
llce	13
llce.object	15
llcelist.object	16
labels.llce	17
merge.formula	18
plot.llcelist	18
predict.gllce	19
predict.llce	20
print.llce	21
Prostate	22
qr.rtr.inv	23
residuals.gllce	24
residuals.llce	25
summary.gllce	25
summary.llce	26
tr	28
vcov.llce	29
Index	30

 aux

Extract Auxiliary Information From an Object

Description

Generic function for extracting auxiliary information from fitted model objects.

Usage

```
aux(object, ...)
```

Arguments

`object` fitted model object (here typically of class `llcelist`, see [aux.llcelist](#)).
`...` potentially further arguments passed to methods.

Details

See documentation (technical reports).

Value

a matrix with the bound(s) (relative [if used] and absolute) and the Lagrangian(s) for the fitted model(s).

aux.l1celist *Use 'aux()' on a 'l1celist' object*

Description

This is a method for the function `aux()` for objects inheriting from class `l1celist`. See [aux](#) for the general behavior of this function and for the interpretation of `object`.

Usage

```
## S3 method for class 'l1celist':
aux(object, ...)
```

Arguments

`object` fitted model object (here typically of class `l1celist`).

`...` potentially further arguments passed to methods.

coef.l1lce *Coefficients of an 'l1lce' Object*

Description

This is a method for `coef()` for objects inheriting from class `l1lce`. See [coef](#) for the general behavior of this function and for the interpretation of `object`.

Usage

```
## S3 method for class 'l1lce':
coef(object, all=TRUE, constrained=FALSE, ...)
```

Arguments

`object` an object of class `l1lce`, see help on [l1lce.object](#).

`all` logical; if false, then only the non-zero coefficients are returned.

`constrained` logical; if true, then only the coefficients that were constrained are returned.

`...` possibly further arguments (none at the moment).

`coef.llcelist` *Coefficients of an 'llcelist' Object*

Description

This is a method for `coef()` for objects inheriting from class `llcelist`. See `coef` for the general behavior of this function and for the interpretation of `object`.

Usage

```
## S3 method for class 'llcelist':
coef(object, all=TRUE, constrained=FALSE, ...)
```

Arguments

<code>object</code>	an object of class <code>llcelist</code> , see help on <code>llcelist.object</code> .
<code>all</code>	logical; if false, then the coefficients that are zero in all fitted models of the list are not returned.
<code>constrained</code>	logical; if true, then only the coefficients that were constrained are returned.
<code>...</code>	possibly further arguments (none at the moment).

`deviance.gllce` *Deviance Method for 'gllce' Objects*

Description

This is a method of the generic function `deviance()` for objects inheriting from class `gllce` (see help(`gllce.object`)).

Usage

```
## S3 method for class 'gllce':
deviance(object, ...)
```

Arguments

<code>object</code>	an object inheriting from class <code>gllce</code> .
<code>...</code>	possibly further arguments (none at the moment).

See Also

`deviance` for the general behavior of this function and for the interpretation of `object`.

deviance.l1ce *Deviance Method for 'l1ce' and 'l1celist' Objects*

Description

These are methods of the generic function `deviance()` for objects inheriting from class `l1ce` or `l1celist` (see `help(l1ce.object)` and `help(l1celist.object)`).

Usage

```
## S3 method for class 'l1ce':
deviance(object, ...)
## S3 method for class 'l1celist':
deviance(object, ...)
```

Arguments

`object` an object inheriting from class `l1ce` or `l1celist`, respectively.
`...` possibly further arguments (none at the moment).

See Also

[deviance](#) for the general behavior of this function and for the interpretation of `object`.

Extract.l1celist *Extract Parts of a 'l1celist' Object*

Description

Allows the user to extract values from a `l1celist` object by using subscripts.

Usage

```
## S3 method for class 'l1celist':
x[..., drop = TRUE]
## S3 method for class 'l1celist':
x[[..., drop = TRUE]]
```

Arguments

`x` an object inheriting from class `"l1celist"`.
`...` a specification of indices – see [Extract](#).
`drop` logical defaulting to `TRUE`. If only one model is subscribed, then it is returned as an object of class `"l1ce"`. If `drop=F`, then an object of class `"l1celist"` is always returned.

Value

an object of class "l1celist" or class "l1ce" extracted from the original list.

fitted.l1ce	<i>Fitted Values for 'l1ce', 'l1celist' and 'gllce' Objects</i>
-------------	---

Description

These are methods of the generic function `fitted()` for objects inheriting from class `l1ce` or `l1celist` (see `help(l1ce.object)` and `help(l1celist.object)`).

Usage

```
## S3 method for class 'l1ce':
fitted(object, ...)
## S3 method for class 'l1celist':
fitted(object, ...)
```

Arguments

<code>object</code>	an object inheriting from class <code>l1ce</code> or <code>l1celist</code> , e.g., an <code>gllce</code> one (see <code>gllce.object</code>).
<code>...</code>	further potential arguments passed to methods.

See Also

`fitted` for the general behavior of this function and for the interpretation of `object`.

<code>gcv</code>	<i>Generalised Cross-Validation Score</i>
------------------	---

Description

Extracts the generalised cross-validation score(s) from fitted model objects.

Usage

```
gcv(object, ...)
```

Arguments

<code>object</code>	fitted model object; see <code>gcv</code> methods for details.
<code>...</code>	arguments passed to methods.

Details

See documentation.

Value

A vector (or matrix) with the bound(s) (relative [if used] and absolute), the Lagrangian(s) and the generalised cross-validation score(s) for the fitted model(s).

gcv.l1ce

'gcv()' Methods for 'l1ce' and 'l1celist' Objects.

Description

This is a method for the function `gcv()` for objects inheriting from class `l1ce` or `l1celist`.

Usage

```
## S3 method for class 'l1ce':
gcv(object, type = c("OPT", "Tibshirani"),
     gen.inverse.diag = 0, ...)
## S3 method for class 'l1celist':
gcv(object, type = c("OPT", "Tibshirani"),
     gen.inverse.diag = 0, ...)
```

Arguments

<code>object</code>	an object of class <code>l1ce</code> or <code>l1celist</code> .
<code>type</code>	character (string) indicating whether to use the covariance formula of Osborne, Presnell and Turlach or the formula of Tibshirani.
<code>gen.inverse.diag</code>	if Tibshirani's formula for the covariance matrix is used, this value is used for the diagonal elements of the generalised inverse that appears in the formula that corresponds to parameters estimated to be zero. The default is 0, i.e. use the Moore-Penrose inverse. Tibshirani's code uses <code>gen.inverse.diag = 1e11</code> .
<code>...</code>	further potential arguments passed to methods.

Details

See documentation.

See Also

[gcv](#) for the general behavior of this function; [l1ce.object](#) and [l1celist.object](#) for description of the `object` argument.

gl1ce

*Generalized Regression With L1-constraint on the Parameters***Description**

Fit a generalized regression problem while imposing an L1 constraint on the parameters. Returns an object of class `gl1ce`.

Usage

```
gl1ce(formula, data = sys.parent(), weights, subset, na.action,
      family = gaussian, control = glm.control(...), sweep.out = ~ 1,
      x = FALSE, y = TRUE, contrasts = NULL, standardize = TRUE,
      guess.constrained.coefficients = double(p), bound = 0.5, ...)
## S3 method for class 'gl1ce':
family(object, ...)
```

Arguments

<code>formula</code>	a formula , with the response on the left hand side of a <code>~</code> operator, and the terms, separated by a <code>+</code> operator, on the right hand side.
<code>data</code>	a data.frame in which to interpret the variables named in the formula, the <code>weights</code> , the <code>subset</code> and the <code>sweep.out</code> argument. If this is missing, then the variables in the formula should be globally available.
<code>weights</code>	vector of observation weights. The length of <code>weights</code> must be the same as the number of observations. The weights must 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 be 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.omit</code> , which deletes observations that contain one or more missing values.
<code>family</code>	a family object - a list of functions and expressions for defining the link and variance functions, initialization and iterative weights. Families supported are <code>gaussian</code> , <code>binomial</code> , <code>poisson</code> , <code>Gamma</code> , <code>inverse.gaussian</code> and <code>quasi</code> . Functions like <code>binomial</code> produce a family object, but can be given without the parentheses. Family functions can take arguments, as in <code>binomial(link=probit)</code> .
<code>control</code>	a list of iteration and algorithmic constants. See <code>glm.control</code> for their names and default values. These can also be set as arguments to <code>gl1ce</code> itself.

sweep.out	a formula object, variables whose parameters are not put under the constraint are swept out first. The variables should appear on the right of a ~ operator and be separated by + operators. Default is ~1, i.e. the constant term is not under the constraint. If this parameter is NULL, then all parameters are put under the constraint.
x	logical flag: if TRUE, the model matrix is returned in component x.
y	logical flag: if TRUE, the response is returned in component y.
contrasts	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.
standardize	logical flag: if TRUE, then the columns of the model matrix that correspond to parameters that are constrained are standardized to have empirical variance one. The standardization is done after taking possible weights into account and after sweeping out variables whose parameters are not constrained.
guess.constrained.coefficients	initial guess for the parameters that are constrained.
bound	numeric, either a single number or a vector: the constraint(s) that is/are put onto the L1 norm of the parameters.
...	potential arguments for <code>glm.control</code> , as default for the <code>control</code> argument above.
object	an R object of class "gllce".

Value

an object of class `gllce` is returned by `gllce()`. See `gllce.object` for details.

References

See the references in `llce`.

Justin Lokhorst (1999). The LASSO and Generalised Linear Models, Honors Project, Nov.1999, Dept.Statist., Univ. of Adelaide. Available as file 'Doc/justin.lokhorst.ps.gz' in both shar files from <http://www.maths.uwa.edu.au/~berwin/software/lasso.html>.

See Also

`glm` for unconstrained generalized regression modeling.

Examples

```
## example from base:
data(esoph)
summary(esoph)
## effects of alcohol, tobacco and interaction, age-adjusted
modEso <- formula(cbind(ncases, ncontrols) ~ agegp + tobgp * alcgp)
glm.E <- glm(modEso, data = esoph, family = binomial())
```

```

gllc.E <- gllce(modEso, data = esoph, family = binomial())
gllc.E
plot(residuals(gllc.E) ~ fitted(gllc.E))

sglc <- summary(gllc.E)
sglc

## Another comparison glm() / gllc.E:
plot(predict(glm.E, type="link"), predict(glm.E, type="response"),
      xlim = c(-3,0))
points(predict(gllc.E, type="link"), predict(gllc.E, type="response"),
       col = 2, cex = 1.5)

labels(gllc.E) #-- oops! empty!!

```

gllce.object

Generalized L1 Constrained Estimation Model Object

Description

These are objects of class `gllce`. They represent the fit of a generalized regression model under an L1 constraint on (some of) the parameters.

Details

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

GENERATION

This class of objects is returned from the `gllce` function to represent a fitted model.

METHODS

The `gllce` class of objects has methods for the following generic functions: `deviance`, `predict`, `print`, `residuals`, `summary`. Other generic functions are inherited from the class `l1ce`.

STRUCTURE

The following components must be included in a legitimate `gllce` object.

coefficients the coefficients of the fit of the response to the columns of the model matrix. The names of the coefficients are the names of the columns of the model matrix.

residuals the residuals from the fit. If weights were used, then the residuals are the raw residuals - the weights are not taken into account. If you need residuals that all have the same variance, then use the `residuals` function with `type="pearson"`.

fitted.values the fitted values from the fit. If weights were used, the fitted values are not adjusted for the weights.

- family** the family of which the fitted regression model belongs, eg., `binomial(link=probit)`.
- bound** the (absolute) L1 constraint imposed on the parameters.
- Lagrangian** the value of the Lagrangian that enforces the constraint at the solution.
- xtx** the moment matrix of the variables that are under the constraint. (After taking weights, sweep-out variables and standardization into account).
- xtr** the product of the design matrix of the variables that are under the constraint (after taking weights, sweep-out variables and standardization into account) with the residual vector.
- constrained.coefficients** the coefficients on the scale on which they are constrained. Useful as initial value for further fits.
- sweep.out** information on the variables that are not under the constraint and on which the other variables and the response is projected first. Optional, not present if `sweep.out = NULL`.
- 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. See also `R.assign` below.
- terms** an object of mode `expression` and class `term` summarizing the formula. Used by various methods, but typically not of direct relevance to users.
- 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.
- 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.
- x** optionally the model matrix, if `x=T`.
- y** optionally the response, if `y=T`.

See Also

[gllce](#), [coefficients](#).

Iowa

The Iowa Wheat Yield Data

Description

The data gives the pre-season and three growing months' precipitation, the mean temperatures for the three growing months and harvest month, the year, and the yield of wheat for the USA state of Iowa, for the years 1930–1962.

Usage

```
data(Iowa)
```

Format

The data frame has the following components:

Year Year of measurement (surrogate for variety improvements)

Rain0 Pre-season rainfall (in.)

Temp1 Mean temperature for the first growing month (deg. F)

Rain1 Rainfall for the first growing month (in.)

Temp2 Mean temperature for the second growing month (deg. F)

Rain2 Rainfall for the second growing month (in.)

Temp3 Mean temperature for the third growing month (deg. F)

Rain3 Rainfall for the third growing month (in.)

Temp4 Mean temperature for the harvest month (deg. F)

Yield Yield of wheat in Iowa for the given year (bush./acre)

CATEGORY

Multiple regression; diagnostics.

Source

CAED Report, 1964. Quoted in Draper and Smith, Applied Regression Analysis.

Examples

```
data(Iowa)
pairs(Iowa)
```

is.formula

Tests for Formula Objects

Description

is.formula returns TRUE if x is an object of class "formula", and FALSE otherwise.

Usage

```
is.formula(x)
```

Arguments

x an R object to be tested.

Description

Returns an object of class "l1ce" or "licelist" that represents fit(s) of linear models while imposing L1 constraint(s) on the parameters.

Usage

```
l1ce(formula, data = sys.parent(), weights, subset, na.action,
     sweep.out = ~ 1, x = FALSE, y = FALSE,
     contrasts = NULL, standardize = TRUE,
     trace = FALSE, guess.constrained.coefficients = double(p),
     bound = 0.5, absolute.t = FALSE)
```

Arguments

formula	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.
data	a <code>data.frame</code> in which to interpret the variables named in the formula, the <code>weights</code> , the <code>subset</code> and the <code>sweep.out</code> argument. If this is missing, then the variables in the formula should be globally available.
weights	vector of observation weights. 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.
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 <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.omit</code> , which deletes observations that contain one or more missing values.
sweep.out	a formula object, variables whose parameters are not put under the constraint are swept out first. The variables should appear on the right of a <code>~</code> operator and be separated by <code>+</code> operators. Default is <code>~1</code> , i.e. the constant term is not under the constraint. If this parameter is <code>NULL</code> , then all parameters are put under the constraint.
x	logical indicating if the model matrix should be returned in component <code>x</code> .
y	logical indicating if the response should be returned in component <code>y</code> .

contrasts	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.
standardize	logical flag: if TRUE, then the columns of the model matrix that correspond to parameters that are constrained are standardized to have empirical variance one. The standardization is done after taking possible weights into account and after sweeping out variables whose parameters are not constrained; see vignette for details.
trace	logical flag: if TRUE, then the status during each iteration of the fitting is reported.
guess.constrained.coefficients	initial guess for the parameters that are constrained.
bound	numeric, either a single number or a vector: the constraint(s) that is/are put onto the L1 norm of the parameters.
absolute.t	logical flag: if TRUE, then bound is an absolute bound and all entries in bound can be any positive number. If FALSE, then bound is a relative bound and all entries must be between 0 and 1; see vignette for details.

Value

an object of class `l1ce` (if `bound` was a single value) or `l1celist` (if `bound` was a vector of values) is returned. See `l1ce.object` and `l1celist.object` for details.

References

Osborne, M.R., Presnell, B. and Turlach, B.A. (2000) On the LASSO and its Dual, *Journal of Computational and Graphical Statistics* **9**(2), 319–337.

Tibshirani, R. (1996) Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society, Series B* **58**(1), 267–288.

Examples

```
data(Iowa)
l1c.I <- l1ce(Yield ~ ., Iowa, bound = 10, absolute.t=TRUE)
l1c.I

## The same, printing information in each step:
l1ce(Yield ~ ., Iowa, bound = 10, trace = TRUE, absolute.t=TRUE)

data(Prostate)
l1c.P <- l1ce(lpsa ~ ., Prostate, bound=(1:30)/30)
length(l1c.P) # 30 l1ce models
l1c.P # -- MM: too large; should do this in summary(.)!

plot(resid(l1c.I) ~ fitted(l1c.I))
abline(h = 0, lty = 3, lwd = .2)
```

l1ce.object

*L1 Constrained Estimation Model Object***Description**

These are objects of class "l1ce". They represent the fit of a regression model under an L1 constraint on (some of) the parameters.

Details

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

GENERATION

This class of objects is returned from the `l1ce` function to represent a fitted model.

METHODS

The "l1ce" class of objects has methods for the following generic functions:
`coef`, `deviance`, `fitted`, `formula`, `gcv`, `labels`, `predict`, `print`, `residuals`, `summary`,
`vcov`.

STRUCTURE

The following components must be included in a legitimate `l1ce` object.

coefficients the coefficients of the fit of the response to the columns of the model matrix. The names of the coefficients are the names of the columns of the model matrix.

residuals the residuals from the fit. If weights were used, then the residuals are the raw residuals - the weights are not taken into account. If you need residuals that all have the same variance, then use the `residuals` function with `type="pearson"`.

fitted.values the fitted values from the fit. If weights were used, the fitted values are not adjusted for the weights.

bound the (absolute) L1 constraint imposed on the parameters.

relative.bound optional, the (relative) L1 constraint imposed on the parameters. Present if `absolute.t=F`.

Lagrangian the value of the Lagrangian that enforces the constraint at the solution.

xtx the moment matrix of the variables that are under the constraint. (After taking weights, sweep-out variables and standardization into account).

xtr the product of the design matrix of the variables that are under the constraint (after taking weights, sweep-out variables and standardization into account) with the residual vector.

constrained.coefficients the coefficients on the scale on which they are constrained. Useful as initial value for further fits.

sweep.out Optional information on the variables that are not under the constraint and on which the other variables and the response is projected first. Optional, not present if `sweep.out = NULL`.

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. See also `R.assign` below.

terms an object of mode `expression` and class `term` summarizing the formula. Used by various methods, but typically not of direct relevance to users.

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.

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.

x optionally the model matrix, if `x=TRUE`.

y optionally the response, if `y=TRUE`.

See Also

[l1ce, coefficients.](#)

`l1celist.object` *Object of Several L1 Constrained Estimation Models*

Description

These are objects of class "l1celist". They represent the fits of several regression models under an L1 constraint on (some of the parameters).

Details

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

GENERATION

This class of objects is returned from the `l1ce` function to represent a fitted model.

METHODS

The "l1celist" class of objects has methods for the following generic functions: `[`, `[[]`, `aux`, `coef`, `deviance`, `fitted`, `formula`, `gcv`, `labels`, `plot`, `print`, `residuals`, `vcov`.

STRUCTURE

An object of class `l1celist` is a list of lists. Each component of the list is a list with the information of an object of class `l1ce` that is unique for that information. All shared information is stored as attributes.

Each component of the list must include the following components for it to be a legitimate `l1celist` object.

coefficients the coefficients of the fit of the response to the columns of the model matrix. The names of the coefficients are the names of the columns of the model matrix.

residuals the residuals from the fit. If weights were used, then the residuals are the raw residuals - the weights are not taken into account. If you need residuals that all have the same variance, then use the `residuals` function with `type="pearson"`.

fitted.values the fitted values from the fit. If weights were used, the fitted values are not adjusted for the weights.

bound the (absolute) L1 constraint imposed on the parameters.

relative.bound optional, the (relative) L1 constraint imposed on the parameters. Present if `absolute.t=F`.

Lagrangian the value of the Lagrangian that enforces the constraint at the solution.

xtr the product of the design matrix of the variables that are under the constraint (after taking weights, sweep-out variables and standardization into account) with the residual vector.

constrained.coefficients the coefficients on the scale on which they are constrained. Useful as initial value for further fits.

See Also

[l1ce](#), [coefficients](#).

labels.l1ce

'Labels' Method for 'l1ce' and 'l1celist' Objects

Description

This is a method for the `labels()` function for objects inheriting from class `l1ce` or `l1celist` (see `help(l1ce.object)` and `help(l1celist.object)`). See `labels` or `labels.default` for the general behavior of this function and for the interpretation of `object`.

Usage

```
## S3 method for class 'l1ce':
labels(object, ...)
## S3 method for class 'l1celist':
labels(object, ...)
```

Arguments

`object` fitted model of class "l1ce" or "l1celist", respectively.
`...` potentially further arguments passed to method.

merge.formula	<i>Merge Formula With Right Hand Side of Second Formula</i>
---------------	---

Description

This is method for formulas of the `merge` generic function. Here it is support for the function `l1ce` and not intended to be called directly by users.

Usage

```
## S3 method for class 'formula':
merge(x, y, ...)
```

Arguments

<code>x, y</code>	formulas.
<code>...</code>	potentially further arguments passed to methods.

Examples

```
merge.formula(y ~ x1, ~ x2) ## -> y ~ x1 + x2

f2 <- merge.formula(y ~ x1*x2, z ~ (x2+x4)^3)
f. <- merge.formula(y ~ x1*x2, ~ (x2+x4)^3) # no LHS for 2nd term
f2
stopifnot(f2 == f.)
```

plot.l1celist	<i>Plot Method for 'l1celist' Objects</i>
---------------	---

Description

Plots a `l1celist` object on the current graphics device.

Usage

```
## S3 method for class 'l1celist':
plot(x, plot=TRUE, all=TRUE, constrained=FALSE,
      type = "b", xlab = "bounds", ylab = "coeff | bounds", ...)
```

Arguments

x	fitted model object of class <code>l1celist</code> .
plot	logical; if TRUE a <code>matplot()</code> of all the coefficients in the list against the relative bound (absolute bound if relative is not present) is plotted. Otherwise no plot is done.
all	logical; if FALSE, then only the non-zero coefficients are returned.
constrained	if TRUE then only the coefficients that were constrained are returned.
type, xlab, ylab, ...	further arguments with useful defaults passed to <code>matplot</code> .

Value

A matrix with the bound(s) (relative [if used] and absolute), the Lagrangian(s) and coefficients of the fitted model(s).

Examples

```
data(Prostate)
l1c.P <- l1ce(lpsa ~ ., Prostate, bound=(1:20)/20)
length(l1c.P) # 20 l1ce models
plot(l1c.P)
```

predict.gllce *Prediction Method for a 'gllce' Object*

Description

This is a method for the generic function `predict` for class "`gllce`", typically produced from `gllce()`. When `newdata` is missing, the fitted values are extracted, otherwise returns *new* predictions.

Usage

```
## S3 method for class 'gllce':
predict(object, newdata, type=c("link", "response"),
        se.fit = FALSE, ...)
```

Arguments

object	a fitted <code>gllce</code> object.
newdata	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> .

type	type of predictions, with choices "link" (the default), or "response". The default produces predictions on the scale of the additive predictors, and with <code>newdata</code> missing, <code>predict()</code> is simply an extractor function for this component of a <code>gllce</code> object. If "response" is selected, the predictions are on the scale of the response, and are monotone transformations of the additive predictors, using the inverse link function.
se.fit	logical indicating if standard errors should be returned as well. Not yet available.
...	further potential arguments passed to methods.

Value

a vector of predictions.

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))`.

Examples

```
## start with
example(gllce)
predict(gllc.E, new = esoph[1:7,])# type 'link'
predict(gllc.E, new = esoph[1:7,], type = "response")

## identities / consistency checks :
stopifnot(predict(gllc.E, type = "response") == fitted(gllc.E),
           all.equal(predict(gllc.E)[1:7],
                     as.vector(predict(gllc.E, new = esoph[1:7,]))),
           all.equal(fitted(gllc.E)[1:7],
                     as.vector(predict(gllc.E, new = esoph[1:7,], type = "response"))
           )
```

predict.l1ce *Predict Method for 'l1ce' Objects*

Description

This is a method for the generic function `predict` for class "l1ce", typically produced from `l1ce()`. When `newdata` is missing, the fitted values are extracted, otherwise returns *new* predictions.

Usage

```
## S3 method for class 'l1ce':
predict(object, newdata,
        type = c("response"), se.fit = FALSE, ...)
```

Arguments

object	a fitted l1ce object.
newdata	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 object need be present by name in newdata.
type	currently only "response".
se.fit	logical indicating if standard errors should be returned as well. Not yet available.
...	further potential arguments passed to methods.

Value

a vector of predictions.

Warning

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

MM: Not sure this is true for R

Examples

```
data(Iowa)
l1c.I <- l1ce(Yield ~ ., Iowa, bound = 10, absolute.t=TRUE)
p10 <- predict(l1c.I, newdata = Iowa[10:19,])
stopifnot(all.equal(p10, fitted(l1c.I)[10:19]))
```

print.l1ce

Print Methods for 'l1ce', 'l1celist' and 'gl1ce' Objects

Description

These are methods of the generic `print()` function for objects inheriting from class `l1ce`, `l1celist` or `gl1ce` (see `help(l1ce.object)`, `help(l1celist.object)` or `help(gl1ce.object)`). See `print` or `print.default` for the general behavior of this function.

Usage

```
## S3 method for class 'l1ce':
print(x, ...)
## S3 method for class 'l1celist':
print(x, ...)
## S3 method for class 'gl1ce':
print(x, ...)
```

Arguments

`x` fitted model of class "l1ce" or "l1celist", respectively.
`...` potentially further arguments passed to method.

 Prostate

Prostate Cancer Data

Description

These data come from a study that examined the correlation between the level of prostate specific antigen and a number of clinical measures in men who were about to receive a radical prostatectomy. It is data frame with 97 rows and 9 columns.

Usage

```
data(Prostate)
```

Format

The data frame has the following components:

lcavol log(cancer volume)
lweight log(prostate weight)
age age
lbph log(benign prostatic hyperplasia amount)
svi seminal vesicle invasion
lcp log(capsular penetration)
gleason Gleason score
pgg45 percentage Gleason scores 4 or 5
lpsa log(prostate specific antigen)

Source

Stamey, T.A., Kabalin, J.N., McNeal, J.E., Johnstone, I.M., Freiha, F., Redwine, E.A. and Yang, N. (1989) Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate: II. radical prostatectomy treated patients, *Journal of Urology* **141**(5), 1076–1083.

Examples

```

data(Prostate)
attach(Prostate)
pairs(Prostate, col = 1+svi, pch = gleason - 5,
      main = paste("Prostate data, n = ", nrow(Prostate)))
detach()

l1c.P <- l1ce(lcavol ~ ., data = Prostate)
coef(l1c.P)[coef(l1c.P) != 0] ## only age, lcp, lpsa (+ intercept)
summary(l1c.P)

```

qr.rtr.inv

Reconstruct the Inverse of $R'R$ from a QR Object

Description

From a QR object, compute the inverse matrix which is implicitly (but not explicitly!) used to solve the underlying least squares problem.

Usage

```
qr.rtr.inv(qr)
```

Arguments

qr qröbject, typically resulting from `qr(.)`.

Value

The $p \times p$ matrix $(R'R)^{-1}$ or equivalently, the inverse of $X'X$ (i.e. $t(X) \%*\% X$ in \mathbb{R}).

See Also

[qr](#), [qr.R](#), [backsolve](#).

Examples

```

(h3 <- 1/outer(0:5, 1:3, "+"))
rtr <- qr.rtr.inv(qr(h3))
all.equal(c(rtr %% 1:3), solve(crossprod(h3), 1:3))

```

residuals.glm *Compute Residuals for 'glm' Objects*

Description

Computes one of the four types of residuals available for `glm` objects. This is a method for the function `residuals()` for objects inheriting from class `glm`. As several types of residuals are available for `glm` objects, there is an additional optional argument `type`.

Usage

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

Arguments

<code>object</code>	an object inheriting from class <code>glm</code> representing a fitted model.
<code>type</code>	type of residuals, with choices "deviance", "pearson", "working" or "response"; the first is the default.
<code>...</code>	possibly further arguments (none at the moment).

Value

A numeric vector of residuals. See *Statistical Models in S* for detailed definitions of each type of residual. The sum of squared deviance residuals add up to the deviance. The pearson residuals are standardized residuals on the scale of the response. The working residuals reside on the object, and are the residuals from the final fit. The response residuals are simply $y - \text{fitted}(\text{object})$. The `summary()` method for `glm` objects produces deviance residuals. The residuals component of a `glm` object contains the working residuals.

References

Chambers, J.M., and Hastie, T.J. (1991). *Statistical Models in S*, pp. 204–206.

See Also

`glm` for examples; `fitted.glm` is used for fitted values.

residuals.l1ce *Residuals of 'l1ce' or 'l1celist' Objects*

Description

This is a method for the function `residuals` for objects inheriting from class `l1ce` or `l1celist` (see `help(l1ce.object)` and `help(l1celist.object)`). See `residuals` or `residuals.default` for the general behavior of this function and for the interpretation of `object` and `type`.

Usage

```
## S3 method for class 'l1ce':
residuals(object, type, ...)
## S3 method for class 'l1celist':
residuals(object, type, ...)
```

Arguments

<code>object</code>	an object inheriting from class <code>l1ce</code> representing a fitted model.
<code>type</code>	type of residuals, with choices "deviance", "pearson", "working" or "response"; the first is the default.
<code>...</code>	possibly further arguments (none at the moment).

summary.g1l1ce *Summary Method for Generalized L1 Constrained Regression Models*

Description

Returns (and prints) a summary list for fitted generalized L1 constrained regression models.

Usage

```
## S3 method for class 'g1l1ce':
summary(object, dispersion = NULL, correlation = FALSE, ...)
## S3 method for class 'summary.g1l1ce':
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

<code>object</code>	fitted <code>g1l1ce</code> model object. This is assumed to be the result of some fit that produces an object inheriting from the class <code>g1l1ce</code> , in the sense that the components returned by the <code>g1l1ce()</code> function will be available.
<code>dispersion</code>	prescribed dispersion, see <code>summary.glm</code> .
<code>correlation</code>	logical indicating if the correlation matrix should be returned as well.
<code>x</code>	object of class <code>summary.g1l1ce</code> .
<code>digits, ...</code>	further arguments for the <code>print</code> method.

Details

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

Value

an object of class `summary.gllce` (for which there's a `print` method). It is basically a list with the following components:

<code>call</code>	as contained on object
<code>residuals</code>	the deviance residuals, as produced by <code>residuals(object, type = "deviance")</code> .
<code>coefficients</code>	the coefficients of the model.
<code>family</code>	the family of models to which object belongs, along with the variance and link functions for that model.
<code>bound</code>	the bound used in fitting this model
<code>Lagrangian</code>	the Lagrangian of the model

<code>summary.l1ce</code>	<i>Summary Method for "l1ce" Objects (Regression with L1 Constraint)</i>
---------------------------	--

Description

Returns a summary list for a regression model with an L1 constraint on the parameters. A null value will be returned if printing is invoked.

Usage

```
## S3 method for class 'l1ce':
summary(object, correlation = TRUE,
        type = c("OPT", "Tibshirani"),
        gen.inverse.diag = 0, sigma = NULL, ...)
## S3 method for class 'summary.l1ce':
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

<code>object</code>	fitted model of class "l1ce".
<code>correlation</code>	logical indicating if the correlation matrix for the coefficients should be included in the summary.
<code>type</code>	character string specifying whether to use the covariance formula of Osborne, Presnell and Turlach or the formula of Tibshirani.

<code>gen.inverse.diag</code>	if Tibshirani's formula for the covariance matrix is used, this value is used for the diagonal elements of the generalised inverse that appears in the formula that corresponds to parameters estimated to be zero. The default is 0, i.e. use the Moore-Penrose inverse. Tibshirani's code uses <code>gen.inverse.diag=1e11</code> .
<code>sigma</code>	the residual standard error estimate. If not provided, then it is estimated by the deviance of the model divided by the error degrees of freedom.
<code>x</code>	an R object of class <code>summary.llce</code> .
<code>digits</code>	number of significant digits to use.
<code>...</code>	further potential arguments passed to methods.

Details

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

Value

an object of class `summary.llce` (for which there's a `print` method). It is basically a list with the following components:

<code>correlation</code>	the computed correlation coefficient matrix for the coefficients in the model.
<code>cov.unscaled</code>	the unscaled covariance matrix; i.e. a matrix such that multiplying it by an estimate of the error variance produces an estimated covariance matrix for the coefficients.
<code>df</code>	the number of degrees of freedom for the model and for residuals.
<code>coefficients</code>	a matrix with three columns, containing the coefficients, their standard errors and the corresponding t statistic.
<code>residuals</code>	the model residuals. These are the weighted residuals if weights were given in the model.
<code>sigma</code>	the residual standard error estimate.
<code>terms</code>	the terms object used in fitting this model.
<code>call</code>	the call object used in fitting this model.
<code>bound</code>	the bound used in fitting this model.
<code>relative.bound</code>	the relative bound used in fitting this model (may not be present).
<code>Lagrangian</code>	the Lagrangian of the model.

See Also

[llce](#), [llce.object](#), [summary](#).

Examples

```

data(Prostate)
summary(l1ce(lpsa ~ .,Prostate))

# Produces the following output:
## Not run:
Call:
  l1ce(formula = lpsa ~ ., data = Prostate)
Residuals:
    Min       1Q   Median       3Q      Max
-1.636 -0.4119  0.076  0.452  1.83

Coefficients:
                Value Std. Error Z score Pr(>|Z|)
(Intercept)  0.7285  1.3898      0.5242  0.6002
    lcavol    0.4937  0.0919      5.3711  0.0000
    lweight    0.2682  0.1774      1.5115  0.1307
    age        0.0000  0.0111      0.0000  1.0000
    lbph       0.0093  0.0587      0.1581  0.8744
    svi        0.4551  0.2525      1.8023  0.0715
    lcp        0.0000  0.0947      0.0000  1.0000
    gleason    0.0000  0.1685      0.0000  1.0000
    pgg45     0.0002  0.0046      0.0391  0.9688

Residual standard error: 0.7595 on 88.36 degrees of freedom
The relative L1 bound was      : 0.5
The absolute L1 bound was     : 0.9219925
The Lagrangian for the bound is: 13.05806

Correlation of Coefficients:
      (Intercept)  lcavol lweight    age    lbph    svi    lcp gleason
lcavol  0.1988
lweight -0.4815    -0.2071
age     -0.3938    -0.0603 -0.0974
lbph    0.3629    -0.0201 -0.5165 -0.1303
svi     -0.0624    -0.2273 -0.1442  0.0635  0.0648
lcp     0.0457    -0.4153  0.0598  0.0665  0.0632 -0.3779
gleason -0.7666    -0.2009  0.1163 -0.0774 -0.0617  0.1084 -0.0243
pgg45   0.4988     0.0956 -0.0380 -0.0630 -0.1111 -0.1921 -0.2935 -0.6526
## End(Not run)

```

tr

*Trace of a Matrix***Description**

Calculates the trace of a matrix

Usage

tr(mat)

Arguments

`mat` a square matrix.

Value

The trace of the matrix, i.e. the sum of its diagonal elements, is returned.

Examples

```
tr(cbind(1,1:3,4:2)) # 5
```

`vcov.l1ce`

Variance-Covariance Matrix of 'l1ce' or 'l1celist' Objects

Description

This is a method for the function `vcov()` for objects inheriting from class `l1ce` or `l1celist` (see `help(l1ce.object)` and `help(l1celist.object)`). See `vcov` for the general behavior of this function.

Usage

```
## S3 method for class 'l1ce':
vcov(object, type = c("OPT", "Tibshirani"),
      gen.inverse.diag = 0, ...)
## S3 method for class 'l1celist':
vcov(object, type = c("OPT", "Tibshirani"),
      gen.inverse.diag = 0, ...)
```

Arguments

`object` an object of class `l1ce` or `l1celist`.

`type` character indicating whether to use the covariance formula of Osborne, Presnell and Turlach or the formula of Tibshirani.

`gen.inverse.diag` if Tibshirani's formula for the covariance matrix is used, this value is used for the diagonal elements of the generalised inverse that appears in the formula that corresponds to parameters estimated to be zero. The default is 0, i.e. use the Moore-Penrose inverse. Tibshirani's code uses `gen.inverse.diag = 1e11`.

`...` further potential arguments passed to methods.

Index

- *Topic **algebra**
 - qr.rtr.inv, 22
- *Topic **array**
 - qr.rtr.inv, 22
- *Topic **classes**
 - gllce.object, 9
 - llce.object, 14
 - llcelist.object, 15
- *Topic **datasets**
 - Iowa, 11
 - Prostate, 21
- *Topic **hplot**
 - plot.llcelist, 18
- *Topic **manip**
 - Extract.llcelist, 4
- *Topic **math**
 - tr, 27
- *Topic **methods**
 - gllce.object, 9
 - llce.object, 14
 - llcelist.object, 15
- *Topic **models**
 - coef.llce, 2
 - coef.llcelist, 3
 - deviance.gllce, 3
 - deviance.llce, 4
 - fitted.llce, 5
 - gcv, 6
 - gcv.llce, 6
 - gllce, 7
 - llce, 12
 - predict.gllce, 18
 - predict.llce, 20
 - residuals.gllce, 23
 - residuals.llce, 24
 - summary.gllce, 24
 - vcov.llce, 28
- *Topic **optimize**
 - gllce, 7
 - llce, 12
- *Topic **print**
 - print.llce, 21
- *Topic **regression**
 - coef.llce, 2
 - coef.llcelist, 3
 - gllce, 7
 - gllce.object, 9
 - llce, 12
 - llce.object, 14
 - llcelist.object, 15
 - summary.llce, 25
- *Topic **utilities**
 - aux, 1
 - aux.llcelist, 2
 - is.formula, 12
 - labels.llce, 17
 - merge.formula, 17
 - [.llcelist(*Extract.llcelist*), 4
 - [[.llcelist(*Extract.llcelist*), 4
 - aux, 1, 2
 - aux.llcelist, 2, 2
 - backsolve, 22
 - binomial, 10
 - coef, 2, 3
 - coef.llce, 2
 - coef.llcelist, 3
 - coefficients, 10, 15, 16
 - data.frame, 7
 - deviance, 3, 4
 - deviance.gllce, 3
 - deviance.llce, 4
 - deviance.llcelist
 - (*deviance.llce*), 4
 - Extract, 5
 - Extract.llcelist, 4

family, 8
family.gllce (*gllce*), 7
fitted, 5
fitted.llce, 5, 23
fitted.llcelist (*fitted.llce*), 5
formula, 7

gcv, 6, 7
gcv.llce, 6
gcv.llcelist (*gcv.llce*), 6
gllce, 7, 10, 18, 19, 23
gllce.object, 3, 5, 8, 9, 21
glm, 9
glm.control, 8

Iowa, 11
is.formula, 12

llce, 9, 12, 14–17, 20, 26
llce.object, 3–5, 7, 13, 14, 17, 21, 24, 26, 28
llcelist.object, 3–5, 7, 13, 15, 17, 21, 24, 28
labels, 17
labels.default, 17
labels.llce, 17
labels.llcelist (*labels.llce*), 17

matplot, 18
merge, 17
merge.formula, 17
model.frame, 13

plot.llcelist, 18
predict, 18, 20
predict.gllce, 18
predict.llce, 20
print, 21
print.default, 21
print.gllce (*print.llce*), 21
print.llce, 21
print.llcelist (*print.llce*), 21
print.summary.gllce
 (*summary.gllce*), 24
print.summary.llce
 (*summary.llce*), 25
Prostate, 21

qr, 22
qr.R, 22
qr.rtr.inv, 22

residuals, 24
residuals.default, 24
residuals.gllce, 23
residuals.llce, 24
residuals.llcelist
 (*residuals.llce*), 24

summary, 26
summary.gllce, 24
summary.glm, 24
summary.llce, 25

tr, 27

vcov, 28
vcov.llce, 28
vcov.llcelist (*vcov.llce*), 28