Package ‘grpreg’

February 20, 2015

Title  Regularization paths for regression models with grouped covariates
Version 2.8-0
Date 2014-11-15
Author Patrick Breheny
Maintainer Patrick Breheny <patrick-breheny@uiowa.edu>
Depends R (>= 2.13.0), Matrix
Description Efficient algorithms for fitting the regularization path of linear or logistic regression models with grouped penalties. This includes group selection methods such as group lasso, group MCP, and group SCAD as well as bi-level selection methods such as the group exponential lasso, the composite MCP, and the group bridge.
License GPL-2
NeedsCompilation yes
Repository CRAN
Date/Publication 2014-11-15 16:54:22

R topics documented:
grpreg-package ....................................................... 2
  birthwt.grpreg ................................................. 3
cv.grpreg ......................................................... 4
gBridge .......................................................... 6
grpreg ............................................................. 8
logLik.grpreg .................................................... 12
plot.cv.grpreg .................................................. 14
plot.grpreg ....................................................... 15
predict.grpreg .................................................. 16
select.grpreg ................................................... 18
summary.cv.grpreg ............................................. 19

Index 21
Description

This package fits regularization paths for linear or logistic regression models with grouped penalties, such as the group lasso, group MCP, group SCAD, and group bridge. The algorithms are based on the idea of either locally approximated coordinate descent or group descent, depending on the penalty. All of the algorithms (with the exception of group bridge) are stable and fast.

Details

The following penalties are available:

- grLasso: Group lasso (Yuan and Lin, 2006)
- grMCP: Group MCP; like the group lasso, but with an MCP penalty on the norm of each group
- grSCAD: Group SCAD; like the group lasso, but with a SCAD penalty on the norm of each group
- cmCP: A hierarchical penalty which places an outer MCP penalty on a sum of inner MCP penalties for each group (Breheny & Huang, 2009)
- gEl: Group exponential lasso
- gBridge: A penalty which places a bridge penalty on the L1-norm of each group (Huang et al., 2009)

The cmCP, gEl, and gBridge penalties carry out bi-level selection, meaning that they carry out variable selection at the group level and at the level of individual covariates (i.e., they select important groups as well as important members of those groups). The grLasso, grMCP, and grSCAD penalties carry out group selection, meaning that within a group, coefficients will either all be zero or all nonzero.

A variety of supporting methods for selecting lambda and plotting the paths are provided also.

See further details at grpreg and gBridge.

The following functions are documented in the help pages:

- grpreg
- gBridge
- cv.grpreg
- logLik.grpreg
- plot.grpreg
- plot.cv.grpreg
- predict.grpreg
- select.grpreg

The following data sets are documented in the help pages:

- birthwt.grpreg
Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>

References

- Breheny, P. The group exponential lasso for bi-level variable selection. In submission.

Examples

```r
## See examples in grpreg
```

---

**birthwt.grpreg**  
*Risk Factors Associated with Low Infant Birth Weight*

Description

The `birthwt.grpreg` data frame has 189 rows and 18 columns. The data were collected at Baystate Medical Center, Springfield, Mass during 1986. This data frame is a reparameterization of the `birthwt` data frame from the `MASS` package.

Usage

`data(birthwt.grpreg)`

Format

This data frame contains the following columns:

- `low` Indicator of birth weight less than 2.5kg
- `bwt` Birth weight in kilograms
- `age1`, `age2`, `age3` Orthogonal polynomials of first, second, and third degree representing mother's age in years
- `lwt1`, `lwt2`, `lwt3` Orthogonal polynomials of first, second, and third degree representing mother's weight in pounds at last menstrual period
- `white`, `black` Indicator functions for mother's race; "other" is reference group
cv.grpreg

- smoke smoking status during pregnancy
- pt1, pt12m Indicator functions for one or for two or more previous premature labors, respectively. No previous premature labors is the reference category.
- ht History of hypertension
- ui Presence of uterine irritability
- ftv1, ftv2, ftv3m Indicator functions for one, for two, or for three or more physician visits during the first trimester, respectively. No visits is the reference category.

Source

MASS. R package. Available at http://cran.r-project.org.

References


Examples

data(birthwt.grpreg)
hist(birthwt.grpreg$bwt, xlab="Child's birth weight", main="")
table(birthwt.grpreg$low)
## See examples in ?birthwt (MASS package)
## for more about the data set
## See examples in grpreg for use of this data set
## with group penalized regression models

cv.grpreg

Cross-validation for grpreg

Description

Performs k-fold cross validation for penalized regression models with grouped covariates over a grid of values for the regularization parameter lambda.

Usage

cv.grpreg(X, y, group=1:ncol(X), ..., nfolds=10, seed, trace=FALSE)
Arguments

\( x \)  
The design matrix, as in `grpreg`.

\( y \)  
The response vector (or matrix), as in `grpreg`.

\( \text{group} \)  
The grouping vector, as in `grpreg`.

\( \ldots \)  
Additional arguments to `grpreg`.

\( \text{n} \text{folds} \)  
The number of cross-validation folds. Default is 10.

\( \text{seed} \)  
You may set the seed of the random number generator in order to obtain reproducible results.

\( \text{trace} \)  
If set to TRUE, cv.grpreg will inform the user of its progress by announcing the beginning of each CV fold. Default is FALSE.

Details

The function calls grpreg \( n \) folds times, each time leaving out \( 1/n \) folds of the data. The cross-validation error is based on the residual sum of squares when `family="gaussian"` and the deviance when `family="binomial"` or `family="poisson"`.

For Gaussian and Poisson responses, the folds are chosen according to simple random sampling. For binomial responses, the numbers for each outcome class are balanced across the folds; i.e., the number of outcomes in which \( y \) is equal to 1 is the same for each fold, or possibly off by 1 if the numbers do not divide evenly.

As in grpreg, seemingly unrelated regressions/multitask learning can be carried out by setting \( y \) to be a matrix, in which case groups are set up automatically (see `grpreg` for details), and cross-validation is carried out with respect to rows of \( y \). As mentioned in the details there, it is recommended to standardize the responses prior to fitting.

Value

An object with S3 class `"cv.grpreg"` containing:

\( \text{cve} \)  
The error for each value of lambda, averaged across the cross-validation folds.

\( \text{cvse} \)  
The estimated standard error associated with each value of for cve.

\( \lambda \)  
The sequence of regularization parameter values along which the cross-validation error was calculated.

\( \text{fit} \)  
The fitted grpreg object for the whole data.

\( \text{min} \)  
The index of lambda corresponding to lambda.min.

\( \lambda \text{.min} \)  
The value of lambda with the minimum cross-validation error.

\( \text{null.dev} \)  
The deviance for the intercept-only model.

\( \text{pe} \)  
If family="binomial", the cross-validation prediction error for each value of lambda.

Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>
Fit a group bridge regression path

gBridge

Description

Fit regularization paths for linear and logistic group bridge-penalized regression models over a grid of values for the regularization parameter lambda.

Usage

gBridge(X, y, group=1:ncol(X), family=c("gaussian", "binomial", "poisson"), nlambdas=100, lambda, lambda.min=(if (nrow(X) > ncol(X)) .001 else .05), lambda.max, alpha=1, eps=.001, delta=1e-7, max.iter=1000, gamma=0.5, group.multiplier=rep(1,J), warn=TRUE)

Arguments

X The design matrix, as in grpreg.
y The response vector (or matrix), as in grpreg.
group The grouping vector, as in grpreg.
family Either "gaussian" or "binomial", depending on the response.
nlambdas The number of lambda values, as in grpreg.
lambda A user supplied sequence of lambda values, as in grpreg.
lambda.min The smallest value for lambda, as in grpreg.
lambda.max The maximum value for lambda. Unlike the penalties in grpreg, it is not possible to solve for lambda.max directly with group bridge models. Thus, it must be specified by the user. If it is not specified, gBridge will attempt a guess at lambda.max, but this is not particularly accurate.

Examples

data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,1:-2])
y <- birthwt.grpreg$bwt
group <- c(1,1,1,2,2,3,3,4,5,6,7,8,8,8)

cvfit <- cv.grpreg(X, y, group)
plot(cvfit)
summary(cvfit)
coef(cvfit) ## Beta at minimum CVE

cvfit <- cv.grpreg(X, y, group, penalty="gel")
plot(cvfit)
summary(cvfit)
alpha Tuning parameter for the balance between the group penalty and the L2 penalty, as in `grpreg`.

delta The group bridge penalty is not differentiable at zero, and requires a small number delta to bound it away from zero. There is typically no need to change this value.

gamma Tuning parameter of the group bridge penalty (the exponent to which the L1 norm of the coefficients in the group are raised). Default is 0.5, the square root.

group.multiplier The multiplicative factor by which each group's penalty is to be multiplied, as in `grpreg`.

Detail
This method fits the group bridge method of Huang et al. (2009). Unlike the penalties in `grpreg`, the group bridge is not differentiable at zero; because of this, a number of changes must be made to the algorithm, which is why it has its own function. Most notably, the method is unable to start at `lambda.max`; it must start at `lambda.max` and proceed in the opposite direction.

In other respects, the usage and behavior of the function is similar to the rest of the `grpreg` package.

Value
An object with S3 class "Bgrpreg", as in `grpreg`.

Author(s)
Patrick Breheny <patrick-breheny@uiowa.edu>

References

See also
`grpreg`

Examples
```r
## Linear regression
data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,,-1:-2])
y <- birthwt.grpreg$bwt
group <- c(1,1,1,2,2,2,3,3,4,5,5,6,7,8,8,8)
```
```r
fit <- gBridge(X, y, group)
plot(fit)
select(fit)

## Logistic regression
y <- birthwt.grpreg$low
fit <- gBridge(X, y, group, family="binomial")
plot(fit)
select(fit)
```

---

**grpreg**

*Fit a group penalized regression path*

**Description**

Fit regularization paths for models with grouped penalties over a grid of values for the regularization parameter lambda. Fits linear and logistic regression models.

**Usage**

```r
gpreg(X, y, group=1:ncol(X), penalty=c("grLasso", "grMCP", "grSCAD", "gel", "cMCP", "gBridge", "glasso", "gmcp"), family=c("gaussian", "binomial", "poisson"), nlambda=100, lambda=lambda.min=(if (nrow(X) > ncol(X)) 1e-4 else .05), alpha=1, eps=.001, max.iter=1000, dfmax=p, gmax=J, gamma=3, tau=1/3, group.multiplier={if (strtrim(penalty,2)=="gr") sqrt(table(group[group!=0])) else rep(1,J)}, warn=TRUE, ...)
```

**Arguments**

- **X**
  The design matrix, without an intercept. `grpreg` standardizes the data and includes an intercept by default.
- **y**
  The response vector, or a matrix in the case of multitask learning (see details).
- **group**
  A vector of describing the grouping of the coefficients. For greatest efficiency and least ambiguity (see details), it is best if `group` is a vector of consecutive integers, although unordered group and named groups are also allowed. If there are coefficients to be included in the model without being penalized, assign them to group 0 (or "0").
- **penalty**
  The penalty to be applied to the model. For group selection, one of `grLasso`, `grMCP`, or `grSCAD`. For bi-level selection, one of `gel` or `cmcp`. See below for details.
- **family**
  Either "gaussian" or "binomial", depending on the response.
- **nlambda**
  The number of lambda values. Default is 100.
- **lambda**
  A user supplied sequence of lambda values. Typically, this is left unspecified, and the function automatically computes a grid of lambda values that ranges uniformly on the log scale over the relevant range of lambda values.
\texttt{lambda.min} 

The smallest value for \( \lambda \), as a fraction of \( \lambda_{\text{max}} \). Default is \(.0001\) if the number of observations is larger than the number of covariates and \(.05\) otherwise.

\texttt{alpha} 

\texttt{grpreg} allows for both a group penalty and an L2 (ridge) penalty; \( \alpha \) controls the proportional weight of the regularization parameters of these two penalties. The group penalties’ regularization parameter is \( \lambda \alpha^{\text{J}} \), while the regularization parameter of the ridge penalty is \( \lambda (1-\alpha) \). Default is \( 1 \): no ridge penalty.

\texttt{eps} 

Convergence threshold. The algorithm iterates until the change (on the standardized scale) in any coefficient is less than \( \epsilon \). Default is \( \text{NPP1} \). See details.

\texttt{max.iter} 

Maximum number of iterations. Default is \( 1000 \). See details.

\texttt{dfmax} 

Limit on the number of parameters allowed to be nonzero. If this limit is exceeded, the algorithm will exit early from the regularization path.

\texttt{gmax} 

Limit on the number of groups allowed to have nonzero elements. If this limit is exceeded, the algorithm will exit early from the regularization path.

\texttt{gamma} 

Tuning parameter of the MCP penalty; defaults to 3.

\texttt{tau} 

Tuning parameter for the group exponential lasso; defaults to \( 1/3 \).

\texttt{group.multiplier} 

A vector of values representing multiplicative factors by which each group’s penalty is to be multiplied. Often, this is a function (such as the square root) of the number of predictors in each group. The default is to use the square root of group size for the group selection methods, and a vector of \( 1 \)’s (i.e., no adjustment for group size) for bi-level selection.

\texttt{warn} 

Should the function give a warning if it fails to converge? Default is \text{TRUE}. See details.

\texttt{...} 

Arguments passed to other functions (such as \texttt{gBridge}).

**Details**

There are two general classes of methods involving grouped penalties: those that carry out bi-level selection and those that carry out group selection. Bi-level means carrying out variable selection at the group level as well as the level of individual covariates (i.e., selecting important groups as well as important members of those groups). Group selection selects important groups, and not members within the group – i.e., within a group, coefficients will either all be zero or all nonzero. The \texttt{grLasso}, \texttt{grMCP}, and \texttt{grSCAD} penalties carry out group selection, while the \texttt{gel} and \texttt{cmcp} penalties carry out bi-level selection. For bi-level selection, see also the \texttt{gBridge} function. For historical reasons and backwards compatibility, some of these penalties have aliases; e.g., \texttt{gLasso} will do the same thing as \texttt{grLasso}, but users are encouraged to use \texttt{grLasso}.

Please note the distinction between \texttt{grMCP} and \texttt{cmcp}. The former involves an MCP penalty being applied to an L2-norm of each group. The latter involves a hierarchical penalty which places an outer MCP penalty on a sum of inner MCP penalties for each group, as proposed in Breheny & Huang, 2009. Either penalty may be referred to as the "group MCP", depending on the publication. To resolve this confusion, Huang et al. (2012) proposed the name "composite MCP" for the \texttt{cmcp} penalty.
For more information about the penalties and their properties, please consult the references below, many of which contain discussion, case studies, and simulation studies comparing the methods. If you use \texttt{grpreg} for an analysis, please cite the appropriate reference.

In keeping with the notation from the original MCP paper, the tuning parameter of the MCP penalty is denoted 'gamma'. Note, however, that in Breheny and Huang (2009), gamma is denoted 'a'.

The objective function is defined to be

\[
\frac{1}{2n} \text{RSS} + \text{penalty}
\]

for "gaussian" and

\[
-\frac{1}{n} \text{loglik} + \text{penalty}
\]

for "binomial", where the likelihood is from a traditional generalized linear model for the log-odds of an event. For logistic regression models, some care is taken to avoid model saturation; the algorithm may exit early in this setting.

For the bi-level selection methods, a locally approximated coordinate descent algorithm is employed. For the group selection methods, group descent algorithms are employed.

The algorithms employed by \texttt{grpreg} are stable and generally converge quite rapidly to values close to the solution. However, especially when \( p \) is large compared with \( n \), \texttt{grpreg} may fail to converge at low values of \( \lambda \), where models are nonidentifiable or nearly singular. Often, this is not the region of the coefficient path that is most interesting. The default behavior warning the user when convergence criteria are not met may be distracting in these cases, and can be modified with \texttt{warn} (convergence can always be checked later by inspecting the value of \texttt{iter}).

If models are not converging, increasing \texttt{max.iter} may not be the most efficient way to correct this problem. Consider increasing \texttt{n.lambda} or \texttt{lambda.min} in addition to increasing \texttt{max.iter}.

Although \texttt{grpreg} allows groups to be unordered and given arbitrary names, it is recommended that you specify groups as consecutive integers. The first reason is efficiency: if groups are out of order, \( X \) must be reordered prior to fitting, then this process reversed to return coefficients according to the original order of \( X \). This is inefficient if \( X \) is very large. The second reason is ambiguity with respect to other arguments such as \texttt{group.multiplier}. With consecutive integers, \texttt{group=3} unambiguously denotes the third element of \texttt{group.multiplier}.

Seemingly unrelated regressions/multitask learning can be carried out using \texttt{grpreg} by passing a matrix to \( y \). In this case, \( X \) will be used in separate regressions for each column of \( y \), with the coefficients grouped across the responses. In other words, each column of \( X \) will form a group with \( m \) members, where \( m \) is the number of columns of \( y \). For multiple Gaussian responses, it is recommended to standardize the columns of \( y \) prior to fitting, in order to apply the penalization equally across columns.

**Value**

An object with S3 class "\texttt{grpreg}" containing:

- \texttt{beta} The fitted matrix of coefficients. The number of rows is equal to the number of coefficients, and the number of columns is equal to \( n \lambda \).
- \texttt{family} Same as above.
- \texttt{group} Same as above.
grpreg

lambda    The sequence of lambda values in the path.
alpha     Same as above.
loss      A vector containing either the residual sum of squares ("gaussian") or negative
          log-likelihood ("binomial") of the fitted model at each value of lambda.
n          Number of observations.
penalty   Same as above.
df        A vector of length n.lambda containing estimates of effective number of model
          parameters all the points along the regularization path. For details on how this
          is calculated, see Breheiny and Huang (2009).
iter      A vector of length n.lambda containing the number of iterations until conver-
          gence at each value of lambda.
group.multiplier
          A named vector containing the multiplicative constant applied to each group’s
          penalty.

Author(s)

Patrick Breheiny <patrick-breheiny@uiowa.edu>

References

pdf
- Huang J., Breheiny, P. and Ma, S. (2012). A selective review of group selection in high dimen-
  Huang2012.pdf
- Breheiny, P. and Huang, J. (to appear) Group descent algorithms for nonconvex penalized linear
- Breheiny, P. The group exponential lasso for bi-level variable selection. In submission. myweb.
  uiowa.edu/pbreheiny/publications/grpexp.pdf

See Also

cv.grpreg, as well as plot and select methods.

Examples

## Linear regression
data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,1:2])
y <- birthwt.grpreg$bwt
group <- c(1,1,1,2,2,2,3,3,4,5,5,6,7,8,8,8)
fit <- grpreg(X,y,group.penalty="grLasso")
plot(fit)
fit <- grpreg(X,y,group.penalty="grMCP")
plot(fit)
fit <- grpreg(X,y,group,penalty="grSCAD")
plot(fit)
fit <- grpreg(X,y,group,penalty="gel")
plot(fit)
fit <- grpreg(X,y,group,penalty="cMCP")
plot(fit)
select(fit,"AIC")

## Logistic regression
y <- birthwt.grpreg$low
fit <- grpreg(X,y,group,penalty="grLasso", family="binomial")
plot(fit)
fit <- grpreg(X,y,group,penalty="grMCP", family="binomial")
plot(fit)
fit <- grpreg(X,y,group,penalty="grSCAD", family="binomial")
plot(fit)
fit <- grpreg(X,y,group,penalty="gel", family="binomial")
plot(fit)
fit <- grpreg(X,y,group,penalty="cMCP", family="binomial")
plot(fit)
select(fit,"BIC")

## Multitask learning
## Simulated example
set.seed(1)
n <- 50
p <- 10
k <- 5
X <- matrix(runif(n*p), n, p)
y <- matrix(rnorm(n*k, X[,1] + X[,2]), n, k)
fit <- grpreg(X, y)
## Note that group is set up automatically:
fit$group
plot(fit)

logLik.grpreg  logLik method for grpreg

Description
Calculates the log likelihood and degrees of freedom for a fitted grpreg object.

Usage
## S3 method for class 'grpreg'
logLik(object, df.method=c("default","active"), REML=FALSE,...)
logLik.grpreg

Arguments

  object       A fitted grpreg object.
  df.method    How should effective model parameters be calculated? One of: "active", which counts the number of nonzero coefficients; or "default", which uses the calculated df returned by grpreg. Default is "default".
  REML         Use restricted MLE for estimation of the scale parameter in a gaussian model? Default is FALSE.
  ...         For S3 method compatibility.

Details

Exists mainly for use with 'AIC' and 'BIC'.

Value

Returns an object of class 'logLik', in this case consisting of a number (or vector of numbers) with two attributes: 'df' (the estimated degrees of freedom in the model) and 'nobs' (number of observations).

The 'print' method for 'logLik' objects is not intended to handle vectors; consequently, the value of the function does not necessarily display correctly. However, it works with 'AIC' and 'BIC' without any glitches and returns the expected vectorized output.

Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>

See Also

grpreg

Examples

data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,-1:-2])
y <- birthwt.grpreg$bwt
group <- c(1,1,1,2,2,2,3,3,4,5,5,5,6,7,8,8,8)
fit <- grpreg(X,y,group,penalty="cMCP")
logLik(fit)  ## Display is glitchy for vectors
AIC(fit)
BIC(fit)
Description

Plots the cross-validation curve from a cv.grpreg object, along with standard error bars.

Usage

```r
## S3 method for class 'cv.grpreg'
plot(x, log.l=TRUE, type=c("cve", "rsq", "scale", "snr", "pred", "all"), selected=TRUE, ...)
```

Arguments

- `x` A cv.grpreg object.
- `log.l` Should horizontal axis be on the log scale? Default is TRUE.
- `type` What to plot on the vertical axis. cve plots the cross-validation error (deviance); rsq plots an estimate of the fraction of the deviance explained by the model (R-squared); snr plots an estimate of the signal-to-noise ratio; scale plots, for family="gaussian", an estimate of the scale parameter (standard deviation); pred plots, for family="binomial", the estimated prediction error; all produces all of the above.
- `selected` If TRUE (the default), places an axis on top of the plot denoting the number of groups in the model (i.e., that contain a nonzero regression coefficient) at that value of lambda.
- `...` Other graphical parameters to `plot`

Details

Error bars representing approximate +/- 1 SE (68% confidence intervals) are plotted along with the estimates at value of lambda. For rsq and snr, these confidence intervals are quite crude, especially near zero, and will hopefully be improved upon in later versions of grpreg.

Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>

See Also

gpreg, cv.grpreg
Examples

```r
## Linear regression
data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,,-1,-2])
y <- birthwt.grpreg$bwt
group <- c(1,1,1,2,2,2,3,3,4,5,5,6,7,8,8,8)
cvfit <- cv.grpreg(X, y, group)
plot(cvfit)
par(mfrow=c(2,2))
plot(cvfit, type="all")

## Logistic regression
y <- birthwt.grpreg$low
cvfit <- cv.grpreg(X, y, group, family="binomial")
plot(cvfit)
par(mfrow=c(2,2))
plot(cvfit, type="all")
```

---

**plot.grpreg**

*Plot coefficients from a "grpreg" object*

### Description

Produces a plot of the coefficient paths for a fitted `grpreg` object.

### Usage

```r
# S3 method for class 'grpreg'
plot(x, alpha=1, legend.loc, log.l=FALSE, norm=FALSE, 
     ...)
```

### Arguments

- `x` Fitted "grpreg" model.
- `alpha` Controls alpha-blending. Default is `alpha=1`.
- `legend.loc` Where should the legend go? If left unspecified, no legend is drawn. See `legend` for details.
- `log.l` Should horizontal axis be on the log scale? Default is `FALSE`.
- `norm` If `TRUE`, plot the norm of each group, rather than the individual coefficients.
- `...` Other graphical parameters to `plot`, `matlines`, or `legend`

### Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>
predict.grpreg

Model predictions based on a fitted grpreg object

Description

Similar to other predict methods, this function returns predictions from a fitted "grpreg" object.

Usage

## S3 method for class 'grpreg'
predict(object, X, type=c("link", "response", "class", "coefficients", "vars", "groups", "nvars", "ngroups", "norm"), lambda, which=1:length(object$lambda), ...)
## S3 method for class 'grpreg'
coef(object, lambda, which=1:length(object$lambda), drop=TRUE, ...)
## S3 method for class 'cv.grpreg'
predict(object, X, lambda=object$lambda.min, which=object$min, type=c("link", "response", "class", "coefficients", "vars", "groups", ",", "nvars", "ngroups", "norm"), ...)
## S3 method for class 'cv.grpreg'
coef(object, lambda=object$lambda.min, which=object$min, ...)
**Arguments**

- **object**: Fitted "grpreg" or "cv.grpreg" model object.
- **x**: Matrix of values at which predictions are to be made. Not used for type="coefficients".
- **lambda**: Values of the regularization parameter lambda at which predictions are requested. For values of lambda not in the sequence of fitted models, linear interpolation is used.
- **which**: Indices of the penalty parameter lambda at which predictions are required. By default, all indices are returned. If lambda is specified, this will override which.
- **type**: Type of prediction: "link" returns the linear predictors; "response" gives the fitted values; "class" returns the binomial outcome with the highest probability; "coefficients" returns the coefficients; "vars" returns the indices for the nonzero coefficients; "groups" returns the indices for the groups with at least one nonzero coefficient; "nvars" returns the number of nonzero coefficients; "ngroups" returns the number of groups with at least one nonzero coefficient; "norm" returns the L2 norm of the coefficients in each group.
- **drop**: By default, if a single value of lambda is supplied, a vector of coefficients is returned. Set drop=FALSE if you wish to have coef always return a matrix (see drop).

**Details**

coe and predict methods are provided for "cv.grpreg" options as a convenience. They simply call coef.grpreg and predict.grpreg with lambda set to the value that minimizes the cross-validation error.

**Value**

The object returned depends on type.

**Author(s)**

Patrick Breheny <patrick-breheny@uiowa.edu>

**See Also**

- `grpreg`

**Examples**

data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,-1:-2])
y <- birthwt.grpreg$low
group <- c(1,1,1,2,2,3,3,4,5,5,6,7,8,8,8)
fit <- grpreg(X, y, group, penalty="grLasso", family="binomial")
coef(fit, lambda=.001)
predict(fit, X, type="link", lambda=.001)
predict(fit, X, type="response", lambda=.001)
predict(fit, X, type="class", lambda=.001)
predict(fit, type="vars", lambda=.07)
predict(fit, type="groups", lambda=.07)
predict(fit, type="norm", lambda=.07)

cvfit <- cv.grpreg(X, y, group, family="binomial", penalty="grMCP")
coef(cvfit)
predict(cvfit, X)
predict(cvfit, type="response")
predict(cvfit, type="groups")

---

**select.grpreg**  
*Select an value of lambda along a grpreg path*

**Description**

Selects a point along the regularization path of a fitted grpreg object according to the AIC, BIC, or GCV criteria.

**Usage**

```r
select(obj,...)  
## S3 method for class 'grpreg'
select(obj, criterion=c("BIC", "AIC", "GCV", "AICc", "EBIC"), df.method=c("default","active"), smooth=FALSE, ...)
```

**Arguments**

- `obj` A fitted grpreg object.
- `criterion` The criterion by which to select the regularization parameter. One of "AIC", "BIC", "GCV", "AICc", or "EBIC"; default is "BIC".
- `df.method` How should effective model parameters be calculated? One of: "active". which counts the number of nonzero coefficients; or "default", which uses the calculated df returned by grpreg. Default is "default".
- `smooth` Applies a smoother to the information criteria before selecting the optimal value.
- `...` For S3 method compatibility.

**Details**

The criteria are defined as follows, where $\ell$ is the log likelihood, $\nu$ is the degrees of freedom, and $n$ is the sample size:

- $AIC = 2\ell + 2\nu$
- $BIC = 2\ell + log(n) \times \nu$
- $GCV = 2\ell / (1 - \nu/n)^2$
- $AICc = AIC + 2 \times \nu / (n - \nu - 1)$
- $EBIC = BIC + 2 \times log(p\text{choosedf})$
Value

A list containing:

- `lambda` The selected value of the regularization parameter, `lambda`.
- `beta` The vector of coefficients at the chosen value of `lambda`.
- `df` The effective number of model parameters at the chosen value of `lambda`.
- `IC` A vector of the calculated model selection criteria for each point on the regularization path.

Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>

See Also

gpreg

Examples

data(birthwt.gpreg)
X <- as.matrix(birthwt.gpreg[, -1:2])
y <- birthwt.gpreg$bwt
group <- c(1, 1, 2, 2, 2, 3, 4, 5, 5, 6, 7, 8, 8, 8)
fit <- grpreg(X, y, group, penalty = "grLasso")
select(fit)
select(fit, crit = "AIC", df = "active")
plot(fit)
abline(v = select(fit)$lambda)
par(mfrow = c(1, 3))
l <- fit$lambda
xlim <- rev(range(l))
plot(l, select(fit)$IC, xlim = xlim, pch = 19, type = "o", ylab = "BIC")
plot(l, select(fit, "AIC")$IC, xlim = xlim, pch = 19, type = "o", ylab = "AIC")
plot(l, select(fit, "GCV")$IC, xlim = xlim, pch = 19, type = "o", ylab = "GCV")
Arguments

- **object**: A "cv.grpreg" object.
- **x**: A "summary.cv.grpreg" object.
- **digits**: Number of digits past the decimal point to print out. Can be a vector specifying different display digits for each of the five non-integer printed values.
- **...**: Further arguments passed to or from other methods.

Value

`summary.cv.grpreg` produces an object with S3 class "summary.cv.grpreg". The class has its own print method and contains the following list elements:

- **penalty**: The penalty used by `grpreg`.
- **model**: Either "linear" or "logistic", depending on the family option in `grpreg`.
- **n**: Number of observations
- **p**: Number of regression coefficients (not including the intercept).
- **min**: The index of `lambda` with the smallest cross-validation error.
- **lambda**: The sequence of `lambda` values used by `cv.grpreg`.
- **cve**: Cross-validation error (deviance).
- **r.squared**: Proportion of variance explained by the model, as estimated by cross-validation.
- **snr**: Signal to noise ratio, as estimated by cross-validation.
- **sigma**: For linear regression models, the scale parameter estimate.
- **pe**: For logistic regression models, the prediction error (misclassification error).

Author(s)

Patrick Breheny <patrick-breheny@uiowa.edu>

See Also

`grpreg`, `cv.grpreg`, `plot.cv.grpreg`

Examples

```r
## Linear regression
data(birthwt.grpreg)
X <- as.matrix(birthwt.grpreg[,1:2])
y <- birthwt.grpreg$bwt
group <- c(1,1,2,2,2,3,3,4,5,5,6,7,8,8,8)
cvfit <- cv.grpreg(X, y, group)
summary(cvfit)

## Logistic regression
y <- birthwt.grpreg$low
cvfit <- cv.grpreg(X, y, group, family="binomial")
summary(cvfit)
```
Index

*Topic datasets
  birthwt.grpreg, 3
*Topic graphics
  plot.cv.grpreg, 14
  plot.grpreg, 15
*Topic models
  gBridge, 6
  grpreg, 8
  birthwt.grpreg, 2, 3
  coef.cv.grpreg (predict.grpreg), 16
  coef.grpreg (predict.grpreg), 16
  cv.grpreg, 2, 4, 11, 14, 20
  drop, 17
  gBridge, 2, 6, 9
  grpreg, 2, 5–7, 8, 14, 20
  grpreg-package, 2
  logLik (loglik.grpreg), 12
  loglik.grpreg, 2, 12
  plot, 11
  plot.cv.grpreg, 2, 6, 14, 20
  plot.grpreg, 2, 15
  predict.cv.grpreg, 6
  predict.cv.grpreg (predict.grpreg), 16
  predict.grpreg, 2, 16
  print.summary.cv.grpreg
    (summary.cv.grpreg), 19
  select, 11
  select (select.grpreg), 18
  select.grpreg, 2, 18
  summary.cv.grpreg, 6, 19