Package ‘GSQR’

March 18, 2018

Type Package

Title Group Penalized Smooth Quantile Regression

Version 1.2

Date 2017-11-17

Author Mohamed Ouhourane, Yi Yang, Karim Oualkacha

Maintainer Mohamed Ouhourane <mohamed.ouhourane@gmail.com>

Description
This package implements a blockwise-majorization-decent for group quantile regression(BMD-SQR), for efficiently computing the solution paths of the group-lasso penalized quantile regression, the group-Mcp penalized quantile regression, the group-Scad penalized quantile regression and theirs approximations.

License GPL-2

NeedsCompilation yes

R topics documented:

GSQR-package *Group Penalized Smooth Quantile Regression*

Description

This package implements a blockwise-majorization-decent for group quantile regression(BMD-SQR), for efficiently computing the solution paths of the group-lasso penalized quantile regression, the group-Mcp penalized quantile regression, the group-Scad penalized quantile regression and theirs approximations.

Details

The DESCRIPTION file:

Package:	GSQR
Type:	Package
Title:	Group Penalized Smooth Quantile Regression
Version:	1.2
Date:	2017-11-17
Author:	Mohamed Ouhourane, Yi Yang, Karim Oualkacha
Maintainer:	Mohamed Ouhourane <mohamed.ouhourane@gmail.com>
Description:	This package implements a blockwise-majorization-decent for group quantile regression(BMD-SQR), f
License:	GPL-2
NeedsCompilation:	yes
Packaged:	2018-03-11 17:12:43 UTC; ouhourane

Index of help topics:

```
GSQR-package            Group Penalized Smooth Quantile Regression
Gsqr                    Fits the regularization paths for group
                        penalized quantile regression
bardet                  Simplified gene expression data from Scheetz et
                        al. (2006)
coef.Gsqr               get coefficients or make coefficient
                        predictions from an "Gsqr" object.
coef.cv.Gsqr            get coefficients or make coefficient
                        predictions from a "cv.Gsqr" object.
cv.Gsqr                 Cross-validation for Gsqr
plot.Gsqr               Plot solution paths from a "Gsqr" object
predict.Gsqr            Outputs predicted responses from an SGL model
                        for new observations
predict.cv.Gsqr         make predictions from a "cv.Gsqr" object.
```

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha

Maintainer: Mohamed Ouhourane <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Mohamed Ouhourane "Group Penalized Smooth Quantile Regression"

---

bardet *Simplified gene expression data from Scheetz et al. (2006)*

Description

Gene expression data (20 genes for 120 samples) from the microarray experiments of mammalian eye tissue samples of Scheetz et al. (2006).

Usage

```
data(bardet)
```

Details

This data set contains 120 samples with 100 predictors (expanded from 20 genes using 5 basis B-splines, as described in Yang, Y. and Zou, H. (2012)).

Value

A list with the following elements:

`x`	a [120 x 100] matrix (expanded from a [120 x 20] matrix) giving the expression levels of 20 filtered genes for the 120 samples. Each row corresponds to a subject, each 5 consecutive columns to a grouped gene.
`y`	a numeric vector of length 120 giving expression level of gene TRIM32, which causes Bardet-Biedl syndrome.

References

Scheetz, T., Kim, K., Swiderski, R., Philp, A., Braun, T., Knudtson, K., Dorrance, A., DiBona, G., Huang, J., Casavant, T. et al. (2006), “Regulation of gene expression in the mammalian eye and its relevance to eye disease”, *Proceedings of the National Academy of Sciences* **103**(39), 14429-14434.

Huang, J., S. Ma, and C.-H. Zhang (2008). “Adaptive Lasso for sparse high-dimensional regression models”. *Statistica Sinica* 18, 1603-1618.

Yang, Y. and Zou, H. (2012), “A Fast Unified Algorithm for Computing Group-Lasso Penalized Learning Problems,” *Statistics and Computing*. Accepted.

Examples

```
# load GSQR library
library(GSQR)

# load data set
data(bardet)

# how many samples and how many predictors ?
dim(bardet$x)

# repsonse y
bardet$y
```

`coef.cv.Gsqr` *get coefficients or make coefficient predictions from a "cv.Gsqr" object.*

Description

This function gets coefficients or makes coefficient predictions from a cross-validated `Gsqr` model, using the stored `"Gsqr"` object, and the optimal value chosen for `lambda`.

Arguments

`cv` fitted `cv.Gsqr` object.

`s` value(s) of the penalty parameter `lambda` at which predictions are required. Default is the value `s="lambda.1se"` stored on the CV `object`, it is the largest value of `lambda` such that error is within 1 standard error of the minimum. Alternatively `s="lambda.min"` can be used, it is the optimal value of `lambda` that gives minimum cross validation error `cvm`.

Details

This function makes it easier to use the results of cross-validation to get coefficients or make coefficient predictions.

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
cv <- cv.Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
coef.cv.Gsqr(cv,s = "lambda.1se")
```

`coef.Gsqr` *get coefficients or make coefficient predictions from an "Gsqr" object.*

Description

Computes the coefficients at the requested values for `lambda` from a fitted `Gsqr` object.

Usage

```
coef.Gsqr(object, s = NULL, ...)
```

Arguments

`object`	fitted `Gsqr` model object.
`s`	value(s) of the penalty parameter `lambda` at which predictions are required. Default is the entire sequence used to create the model.
`...`	not used. Other arguments to predict.

Details

`s` is the new vector at which predictions are requested. If `s` is not in the lambda sequence used for fitting the model, the `coef` function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right `lambda` indices.

Value

The coefficients at the requested values for `lambda`.

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
fit <- Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
coef(fit)
```

cv.Gsqr *Cross-validation for Gsqr*

Description

Does k-fold cross-validation for Gsqr.

Usage

```
cv.Gsqr(x, y, group, Kfold = 5,delta, taux = 0.5, check = c("f1", "f2"), gamm = ifelse(method == "GMcp'
```

Arguments

`x` matrix of predictors, of dimension $N \times p$; each row is an observation vector.

`y` response variable: vector of N observations.

`group` a vector of consecutive integers describing the grouping of the coefficients (see example below).

`Kfold` number of folds - default is 5. Although nfolds can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowable is nfolds=3.

`delta` the parameter δ in the approximation check function . Default is 2.

`taux` quantile to be targeted. Must be in (0,1).

`check` a character string specifying the ckeck function approximation to use, valid options are:

- `"f1"`: the check function 1,
- `"f2"`: the check function 2.

Default is `"f1"`.

`gamm` regularization parameter for the Group Scad or Group Mcp penalty of the coefficients

`method` a character string specifying the function penality to use, valid options are:

- `"GLasso"`: the group Lasso penality,
- `"GScad"`: the group SCAD penality,
- `"GMcp"`: the group MCP penality,
- `"AGScad"`: the approximate group SCAD penality,
- `"AGMcp"`: the approximate group MCP penality,

Default is GLasso.

`plot.it` plot cross-validation the error

`...` other arguments that can be passed to gglasso.

Details

The function runs Gsqr Kfold+1 times; the first to get the lambda sequence, and then the remainder to compute the fit with each of the folds omitted. The average error and standard deviation over the folds are computed.

Value

an object of class `cv.Gsqr` is returned, which is a list with the ingredients of the cross-validation fit.

`cv`	the mean cross-validated error - a vector of length
`cv.error`	estimate of standard error of `cv`.
`finalfit`	An object with S3 class `Gsqr`.
`all.folds`	The folds used in cross-validation.
`lambda`	the values of `lambda` used in the fits.

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
fit <- cv.Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
```

`Gsqr` *Fits the regularization paths for group penalized quantile regression*

Description

Fits a regularization path for the group penalized quantile regression at a sequence of regularization parameters lambda.

Usage

```
Gsqr(x, y, group = NULL, method = c("GLasso", "SGLasso", "GMcp", "GScad", "AGMcp", "AGScad"), check =
```

Arguments

`x`	matrix of predictors, of dimension $N \times p$; each row is an observation vector.
`y`	response variable: vector of N observations.
`group`	a vector of consecutive integers describing the grouping of the coefficients (see example below).
`method`	a character string specifying the function penality to use, valid options are: • `"GLasso"`: the group Lasso penality, • `"GScad"`: the group SCAD penality,

- `"GMcp"`: the group MCP penality,
- `"AGScad"`: the approximate group SCAD penality,
- `"AGMcp"`: the approximate group MCP penality,

Default is `"GLasso"`.

`check` a character string specifying the ckeck function approximation to use, valid options are:

- `"f1"`: the check function 1,
- `"f2"`: the check function 2.

Default is "f1".

`nlambda` the number of `lambda` values - default is 100.

`maxit` maximum number of outer-loop iterations allowed at fixed lambda value. Default is 3e8. If models do not converge, consider increasing `maxit`.

`lambda.factor` The factor for getting the minimal lambda in `lambda` sequence, where `min(lambda)` = `lambda.factor` * `max(lambda)`. `max(lambda)` is the smallest value of `lambda` for which all coefficients are zero. The default depends on the relationship between N (the number of rows in the matrix of predictors) and p (the number of predictors). If $N > p$, the default is `0.0001`, close to zero. If $N < p$, the default is `0.01`. A very small value of `lambda.factor` will lead to a saturated fit. It takes no effect if there is user-defined lambda sequence.

`lambda` a user supplied `lambda` sequence. Typically, by leaving this option unspecified users can have the program compute its own `lambda` sequence based on `nlambda` and `lambda.factor`. Supplying a value of `lambda` overrides this. It is better to supply a decreasing sequence of `lambda` values than a single (small) value, if not, the program will sort user-defined `lambda` sequence in decreasing order automatically.

`pf` penalty factor, a vector in length of bn (bn is the total number of groups). Separate penalty weights can be applied to each group of βs to allow

`dfmax` limit the maximum number of groups in the model. Useful for very large `bs` (group size), if a partial path is desired. Default is `bs+1`.

`pmax` limit the maximum number of groups ever to be nonzero. For example once a group enters the model, no matter how many times it exits or re-enters model through the path, it will be counted only once. Default is `min(dfmax*1.2,bs)`.

`eps` convergence termination tolerance. Defaults value is `1e-8`.

`gamm` regularization parameter for the Group Scad or Group Mcp penalty of the coefficients

`taux` quantile to be targeted. Must be in (0,1).

`delta` the parameter δ in the approximation check function . Default is 2.

`intercept` Whether to include intercept in the model. Default is TRUE.

Value

`b0` intercept sequence of length `length(lambda)`

`beta` a `p*length(lambda)` matrix of coefficients, stored as a sparse matrix class

`lambda` the actual sequence of `lambda` values used

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
fit <- Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
plot(fit)
```

`plot.Gsqr` *Plot solution paths from a "Gsqr" object*

Description

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

Usage

```
## S3 method for class 'Gsqr'
plot(x, group = FALSE, log.l = TRUE, ...)
```

Arguments

`x`	fitted `Gsqr` model
`group`	what is on the Y-axis. Plot the norm of each group if `TRUE`. Plot each coefficient if `FALSE`.
`log.l`	what is on the X-axis. Plot against the log-lambda sequence if `TRUE`. Plot against the lambda sequence if `FALSE`.
`...`	other graphical parameters to plot

Details

A coefficient profile plot is produced.

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
fit <- Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
plot(fit)
```

`predict.cv.Gsqr` *make predictions from a "cv.Gsqr" object.*

Description

This function makes predictions from a cross-validated `Gsqr` model, using the stored `"Gsqr"` object, and the optimal value chosen for `lambda`.

Usage

```
predict.cv.Gsqr(cv, newx, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

`cv` fitted `cv.Gsqr` object.

`newx` matrix of new values for `x` at which predictions are to be made. Must be a matrix. See documentation for `predict.Gsqr`.

`s` value(s) of the penalty parameter `lambda` at which predictions are required. Default is the value `s="lambda.1se"` stored on the CV object. Alternatively `s="lambda.min"` can be used

`...` not used. Other arguments to predict.

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
cv <- cv.Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
predict.cv.Gsqr(cv,bardet$x,s = "lambda.1se")
```

predict.Gsqr *Outputs predicted responses from an SGL model for new observations*

Description

Outputs predicted response values for new user input observations at a specified s value

Usage

```
predict.Gsqr(object, newx, s = NULL)
```

Arguments

object	fitted "GSQR" object
newx	covariate matrix for new observations whose responses we wish to predict
s	the index of the lambda value for the model with which we desire to predict

Value

prediction vector of news

Author(s)

Mohamed Ouhourane, Yi Yang, Karim Oualkacha Maintainer: Mohamed OUHOURANE <mohamed.ouhourane@gmail.com>

References

Mohamed Ouhourane, Yi Yang, Karim Oulkacha: "Group Penalized Smooth Quantile Regression"

Examples

```
data(bardet)
group <- rep(1:20,each=5)
fit <- Gsqr(x=bardet$x,y=bardet$y,group=group,method="GLasso",check="f1",taux=0.5)
predict(fit,bardet$x)
```

Index