# Package: hrqglas (via r-universe)

August 31, 2024

**Type** Package

**Title** Group Variable Selection for Quantile and Robust Mean Regression

**Version** 1.1.0

**Date** 2023-01-29

**Maintainer** Shaobo Li <shaobo.li@ku.edu>

**Description** A program that conducts group variable selection for quantile and robust mean regression (Sherwood and Li, 2022). The group lasso penalty (Yuan and Lin, 2006) is used for group-wise variable selection. Both of the quantile and mean regression models are based on the Huber loss. Specifically, with the tuning parameter in the Huber loss approaching to 0, the quantile check function can be approximated by the Huber loss for the median and the tilted version of Huber loss at other quantiles. Such approximation provides computational efficiency and stability, and has also been shown to be statistical consistent.

**URL** GitHub: https://github.com/shaobo-li/hrqglas

**License** GPL (>= 2)

**Imports** Rcpp (>= 1.0.4), stats, MASS, Matrix, graphics, quantreg

**LinkingTo** Rcpp

**RoxygenNote** 7.1.1

**Encoding** UTF-8

**Repository** https://shaobo-li.r-universe.dev

**RemoteUrl** https://github.com/shaobo-li/hrqglas

**RemoteRef** HEAD

**RemoteSha** 94994e98e572b2b3dcd59c6b478b819579c1bc14

## Contents

---

`coef.cv.hrq_glasso` *Extract coefficients from cv.hrq_glasso object*

---

## Description

Extract coefficients from cv.hrq_glasso object

## Usage

```
## S3 method for class 'cv.hrq_glasso'
coef(object, s, ...)
```

## Arguments

| | |
|---|---|
| `object` | The model object `cv.hrq_glasso` `object`. |
| `s` | Value of lambda. |
| `...` | other input parameters. |

## Value

The function extract estimated coefficients from `cv.hrq_glasso` object.

---

`coef.hrq_glasso` *Extract coefficients from hrq_glasso object*

---

## Description

Extract coefficients from hrq_glasso object

## Usage

```
## S3 method for class 'hrq_glasso'
coef(object, s, ...)
```

## Arguments

| | |
|---|---|
| `object` | The model object `hrq_glasso` `object`. |
| `s` | Value of lambda. |
| `...` | other input parameters. |

## Value

The function extract estimated coefficients from `hrq_glasso` object.

---

`cv.hrq_glasso` *Cross-validation for quantile regression with group lasso*

---

## Description

K fold cross-validation is conducted. Four types of loss (squared error (`se`), absolute error (`ae`) quantile check loss (`check`) and huber loss (`he`)) can be specified as the CV criterion.

## Usage

```
cv.hrq_glasso(
  x,
  y,
  group.index,
  tau = 0.5,
  k = 5,
  loss = "check",
  method = "quantile",
  folds = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | Design matrix |
| `y` | Response variable |
| `group.index` | A vector of group index, e.g., (1,1,1,2,2,2,3,3) |
| `tau` | Percentage |
| `k` | Number of folders. |
| `loss` | The loss function used for computing the cross-validation error. Supported losses include squared error (`se`), absolute error (`ae`), quantile check loss (`check`) and huber loss (`he`). |
| `method` | Choice for mean or quantile regression. Default is `quantile`. |
| `folds` | A vector of folder index for all observations. The procedure random splits if this argument is not specified. |
| `...` | Other inputs of function `hrq_glasso()`. |

## Value

The full solution path is returned. It also returns the vector of CV score as well as the optimal values in terms of min and 1se. Corresponding lambda values are also returned.

| | |
|---|---|
| `beta` | The estimated coefficients for all lambdas, stored in sparse matrix format, where each column corresponds to a lambda. |
| `lambda` | The sequence of lambdas. |
| `lambda.min` | The optimal lambda that minimizes the CV error |
| `lambda.1se` | The largest lambda such that CV error is within 1 standard error of the minimum CV error. |
| `cv.all` | The vector of all values of CV error for all lambdas. |
| `cv.min` | The value of CV error corresponding to `lambda.min`. |
| `cv.1se` | The value of CV error corresponding to `lambda.1se`. |
| `folds` | The vector of indices for k folds split. |
| `cvup` | CV error + 1 standard error |
| `cvlo` | CV error + 1 standard error |
| `n.grp` | The number of selected groups for each lambda. |

## Examples

```
n<- 100
p<- 10
x0<- matrix(rnorm(n*p),n,p)
X<- cbind(x0, x0^2, x0^3)[,order(rep(1:p,3))]
y<- -2+X[,1]+0.5*X[,2]-X[,3]-0.5*X[,7]+X[,8]-0.2*X[,9]+rt(n,2)
group<- rep(1:p, each=3)
fitcv<- cv.hrq_glasso(x=X, y=y, group.index=group, method="quantile")
plot(fitcv)
```

---

`hrq_glasso` *Robust group variable selection for quantile and mean regression*

---

## Description

This function conducts group-wise (with known groups) variable selection for quantile and robust mean regression with the group lasso penalty. The Huber loss is used for both types of regression model, where the quantile check function is approximated by Huber loss. A full solution path is generated unless a single value of the shrinkage parameter is specified.

## Usage

```
hrq_glasso(
  x,
  y,
  group.index,
  tau = 0.5,
  lambda = NULL,
  weights = NULL,
  w.lambda = NULL,
  gamma = 0.2,
  max_iter = 200,
  apprx = "huber",
  lambda.discard = TRUE,
  method = "quantile",
  scalex = TRUE,
  epsilon = 1e-04,
  beta0 = NULL
)
```

## Arguments

| | |
|---|---|
| x | Design matrix (in matrix format) |
| y | Response variable |
| group.index | A vector of group index, e.g., (1,1,1,2,2,2,3,3) |
| tau | Percentile |
| lambda | Shrinkage parameter, default is NULL so that the algorithm chooses a sequence. |
| weights | Observation weights, default is NULL |
| w.lambda | Weights for Shrinkage parameter of each group, default is NULL |
| gamma | Huber parameter. An initial value is 0.2, while the algorithm adaptively tunes the value in each iteration. |
| max_iter | Maximum number of iteration |
| apprx | Approximation method. Default is huber. The other option is tanh which uses the hypertangent function to approximate the first order derivative of absolute loss. |
| lambda.discard | Default is TRUE, meaning that the solution path stops if the relative deviance changes sufficiently small. It usually happens near the end of solution path. However, the program returns at least 70 models along the solution path. |
| method | Choice for mean or quantile regression. Default is quantile. |
| scalex | Standardize design matrix. Default is TRUE. |
| epsilon | The epsilon level convergence. Default is 1e-4. |
| beta0 | Initial estimates. Default is NULL. |

## Value

It returns a sequence of estimated coefficients for quantile regression with group feature selection corresponding to a sequence of lambda. The estimated coefficients are in the sparse matrix format. Returned values also include the sequence of lambda, the null deviance, values of penalized loss, and unpenalized loss across the sequence of lambda.

| | |
|---|---|
| `beta` | The estimated coefficients for all lambdas, stored in sparse matrix format, where each column corresponds to a lambda. |
| `lambda` | The sequence of lambdas. |
| `null.dev` | The null deviance. |
| `pen.loss` | The value of penalized loss for each lambda. |
| `loss` | The value of unpenalized loss for each lambda. |
| `index.grp` | Group indices that correspond to the estimated coefficient matrix `beta`. |
| `n.grp` | The number of selected groups for each lambda. |

## References

Sherwood, B., and Li, S. (2021) An Efficient Approach to Feature Selection and Estimation for Quantile Regression with Grouped Variables. *Working paper*.

Yang, Y., and Zou, H., (2015) A Fast Unified Algorithm for Solving Group-lasso Penalize Learning Problems, *Statistics and Computing*, 25 1129-1141. doi:10.1007/s1122201494985.

## Examples

```
n<- 100
p<- 10
x0<- matrix(rnorm(n*p),n,p)
X<- cbind(x0, x0^2, x0^3)[,order(rep(1:p,3))]
y<- -2+X[,1]+0.5*X[,2]-X[,3]-0.5*X[,7]+X[,8]-0.2*X[,9]+rt(n,2)
group<- rep(1:p, each=3)
fit<- hrq_glasso(X, y, group)
fit$beta[,8]
```

---

# plot.cv.hrq_glasso *Generating plots for cross-validation*

---

## Description

Generating plots for cross-validation

## Usage

```
## S3 method for class 'cv.hrq_glasso'
plot(x, ...)
```

### Arguments

| | |
|---|---|
| x | The object of function cv.hrq_glasso. |
| ... | other input parameters for the generic function plot. |

### Value

Cross-validation plot for the entire solution path.

---

## predict.cv.hrq_glasso *Prediction for cv.hrq_glasso object*

---

### Description

Prediction for cv.hrq_glasso object

### Usage

```
## S3 method for class 'cv.hrq_glasso'
predict(object, newX, s, ...)
```

### Arguments

| | |
|---|---|
| object | The model object of cv.hrq_glasso. |
| newX | New design matrix. |
| s | Value of lambda. If missing, the default is the lambda.min. |
| ... | other input parameters. |

### Value

The function returns predicted values based on the fitted model from cv.hrq_glasso.

---

## predict.hrq_glasso *Prediction for the hrq_glasso object*

---

### Description

This function provides the prediction of the hrq_glasso object.

### Usage

```
## S3 method for class 'hrq_glasso'
predict(object, newX, s = NULL, ...)
```

## Arguments

| | |
|---|---|
| object | The model object of hrq_glasso. |
| newX | New design matrix. |
| s | Value of lambda. Default is NULL, so that the function provides prediction at all lambdas used in hrq_glasso. |
| ... | other input parameters. |

## Value

The function returns predicted values based on the fitted model from hrq_glasso.

## Examples

```
n<- 100
p<- 10
x0<- matrix(rnorm(n*p),n,p)
X<- cbind(x0, x0^2, x0^3)[,order(rep(1:p,3))]
y<- -2+X[,1]+0.5*X[,2]-X[,3]-0.5*X[,7]+X[,8]-0.2*X[,9]+rt(n,2)
group<- rep(1:p, each=3)
fit<- hrq_glasso(X, y, group)
pred<- predict(fit, newX=X, s=0.3)
```

# Index