Package ‘conquer’

November 1, 2021

Type Package

Title Convolution-Type Smoothed Quantile Regression

Version 1.2.1

Date 2021-10-31

Description Estimation and inference for conditional linear quantile regression models using a convolution smoothed approach. In the low-dimensional setting, efficient gradient-based methods are employed for fitting both a single model and a regression process over a quantile range. Normal-based and (multiplier) bootstrap confidence intervals for all slope coefficients are constructed. In high dimensions, the conquer methods complemented with l_1-penalization and iteratively reweighted l_1-penalization are used to fit sparse models.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

URL `https://github.com/XiaoouPan/conquer`

SystemRequirements C++11

Imports Rcpp (>= 1.0.3), Matrix, matrixStats, stats, caret

LinkingTo Rcpp, RcppArmadillo (>= 0.9.850.1.0)

RoxygenNote 7.1.1

NeedsCompilation yes

Author Xuming He [aut],
Xiaoou Pan [aut, cre],
Kean Ming Tan [aut],
Wen-Xin Zhou [aut]

Maintainer Xiaoou Pan <`xip024@ucsd.edu`>

Repository CRAN

Date/Publication 2021-11-01 15:50:04 UTC

R topics documented:

`conquer-package` *Conquer: Convolution-Type Smoothed Quantile Regression*

Description

Estimation and inference for conditional linear quantile regression models using a convolution smoothed approach. In the low-dimensional setting, efficient gradient-based methods are employed for fitting both a single model and a regression process over a quantile range. Normal-based and (multiplier) bootstrap confidence intervals for all slope coefficients are constructed. In high dimensions, the conquer methods complemented with ℓ_1-penalization and iteratively reweighted ℓ_1-penalization are used to fit sparse models.

Author(s)

Xuming He <xmhe@umich.edu>, Xiaoou Pan <xip024@ucsd.edu>, Kean Ming Tan <keanming@umich.edu>, and Wen-Xin Zhou <wez243@ucsd.edu>

References

Barzilai, J. and Borwein, J. M. (1988). Two-point step size gradient methods. IMA J. Numer. Anal. 8 141–148.

Belloni, A. and Chernozhukov, V. (2011). ℓ_1 penalized quantile regression in high-dimensional sparse models. Ann. Statist. 39 82-130.

Fan, J., Liu, H., Sun, Q. and Zhang, T. (2018). I-LAMM for sparse learning: Simultaneous control of algorithmic complexity and statistical error. Ann. Statist. 46 814-841.

Fernandes, M., Guerre, E. and Horta, E. (2019). Smoothing quantile regressions. J. Bus. Econ. Statist. 39 338-357.

He, X., Pan, X., Tan, K. M., and Zhou, W.-X. (2021+). Smoothed quantile regression for large-scale inference. J. Econometrics, in press.

Horowitz, J. L. (1998). Bootstrap methods for median regression models. Econometrica 66 1327–1351.

Koenker, R. (2005). Quantile Regression. Cambridge University Press, Cambridge.

Koenker, R. Package "quantreg".

Koenker, R. and Bassett, G. (1978). Regression quantiles. Econometrica 46 33-50.

Tan, K. M., Wang, L. and Zhou, W.-X. (2021+). High-dimensional quantile regression: convolution smoothing and concave regularization. J. Roy. Statist. Soc. Ser. B, in press.

`conquer` *Convolution-Type Smoothed Quantile Regression*

Description

Estimation and inference for conditional linear quantile regression models using a convolution smoothed approach. Efficient gradient-based methods are employed for fitting both a single model and a regression process over a quantile range. Normal-based and (multiplier) bootstrap confidence intervals for all slope coefficients are constructed.

Usage

```
conquer(
  X,
  Y,
  tau = 0.5,
  kernel = c("Gaussian", "logistic", "uniform", "parabolic", "triangular"),
  h = 0,
  checkSing = FALSE,
  tol = 1e-04,
  iteMax = 5000,
  ci = FALSE,
  alpha = 0.05,
  B = 1000
)
```

Arguments

`X`	A n by p design matrix. Each row is a vector of observation with p covariates. Number of observations n must be greater than number of covariates p.
`Y`	An n-dimensional response vector.
`tau`	(**optional**) The desired quantile level. Default is 0.5. Value must be between 0 and 1.
`kernel`	(**optional**) A character string specifying the choice of kernel function. Default is "Gaussian". Choices are "Gaussian", "logistic", "uniform", "parabolic" and "triangular".
`h`	(**optional**) Bandwidth/smoothing parameter. Default is $\max\{((log(n)+p)/n)^{0.4}, 0.05\}$. The default will be used if the input value is less than 0.05.
`checkSing`	(**optional**) A logical flag. Default is FALSE. If `checkSing = TRUE`, then it will check if the design matrix is singular before running conquer.
`tol`	(**optional**) Tolerance level of the gradient descent algorithm. The iteration will stop when the maximum magnitude of all the elements of the gradient is less than `tol`. Default is 1e-04.
`iteMax`	(**optional**) Maximum number of iterations. Default is 5000.

`ci`	(**optional**) A logical flag. Default is FALSE. If `ci = TRUE`, then three types of confidence intervals (percentile, pivotal and normal) will be constructed via multiplier bootstrap.
`alpha`	(**optional**) Miscoverage level for each confidence interval. Default is 0.05.
`B`	(**optional**) The size of bootstrap samples. Default is 1000.

Value

An object containing the following items will be returned:

`coeff` A $(p+1)$-vector of estimated quantile regression coefficients, including the intercept.

`ite` Number of iterations until convergence.

`residual` An n-vector of fitted residuals.

`bandwidth` Bandwidth value.

`tau` Quantile level.

`kernel` Kernel function.

`n` Sample size.

`p` Number of covariates.

`perCI` The percentile confidence intervals for regression coefficients. Not available if `ci = FALSE`.

`pivCI` The pivotal confidence intervals for regression coefficients. Not available if `ci = FALSE`.

`normCI` The normal-based confidence intervals for regression coefficients. Not available if `ci = FALSE`.

Author(s)

Xuming He <xmhe@umich.edu>, Xiaoou Pan <xip024@ucsd.edu>, Kean Ming Tan <keanming@umich.edu>, and Wen-Xin Zhou <wez243@ucsd.edu>

References

Barzilai, J. and Borwein, J. M. (1988). Two-point step size gradient methods. IMA J. Numer. Anal. 8 141–148.

Fernandes, M., Guerre, E. and Horta, E. (2019). Smoothing quantile regressions. J. Bus. Econ. Statist., in press.

He, X., Pan, X., Tan, K. M., and Zhou, W.-X. (2021+). Smoothed quantile regression for large-scale inference. J. Econometrics, in press.

Koenker, R. and Bassett, G. (1978). Regression quantiles. Econometrica 46 33-50.

See Also

See `conquer.process` for smoothed quantile regression process.

Examples

```
n = 500; p = 10
beta = rep(1, p)
X = matrix(rnorm(n * p), n, p)
Y = X %*% beta + rt(n, 2)

## Smoothed quantile regression with Gaussian kernel
fit.Gauss = conquer(X, Y, tau = 0.5, kernel = "Gaussian")
beta.hat.Gauss = fit.Gauss$coeff

## Smoothe quantile regression with uniform kernel
fit.unif = conquer(X, Y, tau = 0.5, kernel = "uniform")
beta.hat.unif = fit.unif$coeff

## Construct three types of confidence intervals via multiplier bootstrap
fit = conquer(X, Y, tau = 0.5, kernel = "Gaussian", ci = TRUE)
ci.per = fit$perCI
ci.piv = fit$pivCI
ci.norm = fit$normCI
```

`conquer.cv.reg` *Cross-Validated Penalized Convolution-Type Smoothed Quantile Regression*

Description

Fit sparse quantile regression models via regularized conquer methods with "lasso", "scad" and "mcp" penalties. The regularization parameter λ is selected by cross-validation.

Usage

```
conquer.cv.reg(
  X,
  Y,
  lambdaSeq = NULL,
  tau = 0.5,
  kernel = c("Gaussian", "logistic", "uniform", "parabolic", "triangular"),
  h = 0,
  penalty = c("lasso", "scad", "mcp"),
  kfolds = 5,
  numLambda = 50,
  para = NULL,
  epsilon = 0.001,
  iteMax = 500,
  phi0 = 0.01,
  gamma = 1.2,
  iteTight = 3
)
```

Arguments

`X`	A n by p design matrix. Each row is a vector of observation with p covariates.
`Y`	An n-dimensional response vector.
`lambdaSeq`	(**optional**) A sequence of candidate regularization parameters. If unspecified, the sequence will be generated by a simulated pivotal quantity approach proposed by Belloni and Chernozhukov (2011).
`tau`	(**optional**) Quantile level (between 0 and 1). Default is 0.5.
`kernel`	(**optional**) A character string specifying the choice of kernel function. Default is "Gaussian". Choices are "Gaussian", "logistic", "uniform", "parabolic" and "triangular".
`h`	(**optional**) The bandwidth parameter for kernel smoothing. Default is $\max\{0.5 * (log(p)/n)^{0.25}, 0.05\}$.
`penalty`	(**optional**) A character string specifying the penalty. Default is "lasso". Choices are "lasso", "scad" or "mcp".
`kfolds`	(**optional**) Number of folds for cross-validation. Default is 5.
`numLambda`	(**optional**) Number of λ values for cross-validation if `lambdaSeq` is unspeficied. Default is 50.
`para`	(**optional**) A constant parameter for "scad" and "mcp". Do not need to specify if the penalty is lasso. The default values are 3.7 for "scad" and 3 for "mcp".
`epsilon`	(**optional**) A tolerance level for the stopping rule. The iteration will stop when the maximum magnitude of the change of coefficient updates is less than `epsilon`. Default is 0.001.
`iteMax`	(**optional**) Maximum number of iterations. Default is 500.
`phi0`	(**optional**) The initial quadratic coefficient parameter in the local adaptive majorize-minimize algorithm. Default is 0.01.
`gamma`	(**optional**) The adaptive search parameter (greater than 1) in the local adaptive majorize-minimize algorithm. Default is 1.2.
`iteTight`	(**optional**) Maximum number of tightening iterations in the iteratively reweighted ℓ_1-penalized algorithm. Do not need to specify if the penalty is lasso. Default is 3.

Value

An object containing the following items will be returned:

`coeff` A $(p+1)$ vector of estimated coefficients, including the intercept.

`lambda` Regularization parameter selected by cross-validation.

`bandwidth` Bandwidth value.

`tau` Quantile level.

`kernel` Kernel function.

`penalty` Penalty type.

`n` Sample size.

`p` Number of covariates.

Author(s)

Xuming He <xmhe@umich.edu>, Xiaoou Pan <xip024@ucsd.edu>, Kean Ming Tan <keanming@umich.edu>, and Wen-Xin Zhou <wez243@ucsd.edu>

References

Belloni, A. and Chernozhukov, V. (2011). ℓ_1 penalized quantile regression in high-dimensional sparse models. Ann. Statist. 39 82-130.

Fan, J., Liu, H., Sun, Q. and Zhang, T. (2018). I-LAMM for sparse learning: Simultaneous control of algorithmic complexity and statistical error. Ann. Statist. 46 814-841.

Koenker, R. and Bassett, G. (1978). Regression quantiles. Econometrica 46 33-50.

Tan, K. M., Wang, L. and Zhou, W.-X. (2021). High-dimensional quantile regression: convolution smoothing and concave regularization. J. Roy. Statist. Soc. Ser. B, to appear.

See Also

See `conquer.reg` for regularized quantile regression with a prescribed $lambda$.

Examples

```
n = 100; p = 100; s = 3
beta = c(rep(1.5, s), rep(0, p - s))
X = matrix(rnorm(n * p), n, p)
Y = X %*% beta + rt(n, 2)

## Cross-validated regularized conquer with lasso penalty at tau = 0.8
fit.lasso = conquer.cv.reg(X, Y, tau = 0.8, kernel = "Gaussian", penalty = "lasso")
beta.lasso = fit.lasso$coeff

#' ## Cross-validated regularized conquer with scad penalty at tau = 0.8
fit.scad = conquer.cv.reg(X, Y,tau = 0.8, kernel = "Gaussian", penalty = "scad")
beta.scad = fit.scad$coeff
```

`conquer.process` *Convolution-Type Smoothed Quantile Regression Process*

Description

Fit a smoothed quantile regression process over a quantile range. The algorithm is essentially the same as `conquer`.

Usage

```
conquer.process(
  X,
  Y,
  tauSeq = seq(0.1, 0.9, by = 0.05),
  kernel = c("Gaussian", "logistic", "uniform", "parabolic", "triangular"),
  h = 0,
  checkSing = FALSE,
  tol = 1e-04,
  iteMax = 5000
)
```

Arguments

`X`	A n by p design matrix. Each row is a vector of observation with p covariates. Number of observations n must be greater than number of covariates p.
`Y`	An n-dimensional response vector.
`tauSeq`	(**optional**) A sequence of quantile values (between 0 and 1). Default is $\{0.1, 0.15, 0.2, ..., 0.85, 0.9\}$.
`kernel`	(**optional**) A character string specifying the choice of kernel function. Default is "Gaussian". Choices are "Gaussian", "logistic", "uniform", "parabolic" and "triangular".
`h`	(**optional**) The bandwidth/smoothing parameter. Default is $\max\{((log(n) + p)/n)^{0.4}, 0.05\}$. The default will be used if the input value is less than 0.05.
`checkSing`	(**optional**) A logical flag. Default is FALSE. If `checkSing = TRUE`, then it will check if the design matrix is singular before running conquer.
`tol`	(**optional**) Tolerance level of the gradient descent algorithm. The iteration will stop when the maximum magnitude of all the elements of the gradient is less than `tol`. Default is 1e-04.
`iteMax`	(**optional**) Maximum number of iterations. Default is 5000.

Value

An object containing the following items will be returned:

`coeff` A $(p + 1)$ by m matrix of estimated quantile regression process coefficients, including the intercept. m is the length of `tauSeq`.

`bandwidth` Bandwidth value.

`tauSeq` The sequence of quantile levels.

`kernel` The choice of kernel function.

`n` Sample size.

`p` Number the covariates.

Author(s)

Xuming He <xmhe@umich.edu>, Xiaoou Pan <xip024@ucsd.edu>, Kean Ming Tan <keanming@umich.edu>, and Wen-Xin Zhou <wez243@ucsd.edu>

References

Barzilai, J. and Borwein, J. M. (1988). Two-point step size gradient methods. IMA J. Numer. Anal. 8 141–148.

Fernandes, M., Guerre, E. and Horta, E. (2019). Smoothing quantile regressions. J. Bus. Econ. Statist., in press.

He, X., Pan, X., Tan, K. M., and Zhou, W.-X. (2021+). Smoothed quantile regression for large-scale inference. J. Econometrics, in press.

Koenker, R. and Bassett, G. (1978). Regression quantiles. Econometrica 46 33-50.

See Also

See `conquer` for single-index smoothed quantile regression.

Examples

```
n = 500; p = 10
beta = rep(1, p)
X = matrix(rnorm(n * p), n, p)
Y = X %*% beta + rt(n, 2)

## Smoothed quantile regression process with Gaussian kernel
fit.Gauss = conquer.process(X, Y, tauSeq = seq(0.2, 0.8, by = 0.05), kernel = "Gaussian")
beta.hat.Gauss = fit.Gauss$coeff

## Smoothe quantile regression with uniform kernel
fit.unif = conquer.process(X, Y, tauSeq = seq(0.2, 0.8, by = 0.05), kernel = "uniform")
beta.hat.unif = fit.unif$coeff
```

`conquer.reg` *Penalized Convolution-Type Smoothed Quantile Regression*

Description

Fit sparse quantile regression models in high dimensions via regularized conquer methods with "lasso", "scad" and "mcp" penalties. For "scad" and "mcp", the iteratively reweighted ℓ_1-penalized algorithm is complemented with a local adpative majorize-minimize algorithm.

Usage

```
conquer.reg(
  X,
  Y,
  lambda = 0.2,
  tau = 0.5,
  kernel = c("Gaussian", "logistic", "uniform", "parabolic", "triangular"),
  h = 0,
  penalty = c("lasso", "scad", "mcp"),
  para = NULL,
  epsilon = 0.001,
  iteMax = 500,
  phi0 = 0.01,
  gamma = 1.2,
  iteTight = 3
)
```

Arguments

`X`	A n by p design matrix. Each row is a vector of observation with p covariates.
`Y`	An n-dimensional response vector.
`lambda`	(**optional**) Regularization parameter. Default is 0.2.
`tau`	(**optional**) Quantile level (between 0 and 1). Default is 0.5.
`kernel`	(**optional**) A character string specifying the choice of kernel function. Default is "Gaussian". Choices are "Gaussian", "logistic", "uniform", "parabolic" and "triangular".
`h`	(**optional**) Bandwidth/smoothing parameter. Default is $\max\{0.5*(log(p)/n)^{0.25}, 0.05\}$.
`penalty`	(**optional**) A character string specifying the penalty. Default is "lasso". The other two options are "scad" and "mcp".
`para`	(**optional**) A constant parameter for "scad" and "mcp". Do not need to specify if the penalty is lasso. The default values are 3.7 for "scad" and 3 for "mcp".
`epsilon`	(**optional**) A tolerance level for the stopping rule. The iteration will stop when the maximum magnitude of the change of coefficient updates is less than `epsilon`. Default is 0.001.
`iteMax`	(**optional**) Maximum number of iterations. Default is 500.
`phi0`	(**optional**) The initial quadratic coefficient parameter in the local adaptive majorize-minimize algorithm. Default is 0.01.
`gamma`	(**optional**) The adaptive search parameter (greater than 1) in the local adaptive majorize-minimize algorithm. Default is 1.2.
`iteTight`	(**optional**) Maximum number of tightening iterations in the iteratively reweighted ℓ_1-penalized algorithm. Do not need to specify if the penalty is lasso. Default is 3.

Value

An object containing the following items will be returned:

`coeff` A $(p+1)$ vector of estimated coefficients, including the intercept.

`bandwidth` Bandwidth value.

`tau` Quantile level.

`kernel` Kernel function.

`penalty` Penalty type.

`lambda` Regularization parameter.

`n` Sample size.

`p` Number of the covariates.

Author(s)

Xuming He <xmhe@umich.edu>, Xiaoou Pan <xip024@ucsd.edu>, Kean Ming Tan <keanming@umich.edu>, and Wen-Xin Zhou <wez243@ucsd.edu>

References

Fan, J., Liu, H., Sun, Q. and Zhang, T. (2018). I-LAMM for sparse learning: Simultaneous control of algorithmic complexity and statistical error. Ann. Statist. 46 814-841.

Koenker, R. and Bassett, G. (1978). Regression quantiles. Econometrica 46 33-50.

Tan, K. M., Wang, L. and Zhou, W.-X. (2021). High-dimensional quantile regression: convolution smoothing and concave regularization. J. Roy. Statist. Soc. Ser. B, to appear.

See Also

See `conquer.cv.reg` for regularized quantile regression with cross-validation.

Examples

```
n = 200; p = 500; s = 10
beta = c(rep(1.5, s), rep(0, p - s))
X = matrix(rnorm(n * p), n, p)
Y = X %*% beta + rt(n, 2)

## Regularized conquer with lasso penalty at tau = 0.8
fit.lasso = conquer.reg(X, Y, lambda = 0.05, tau = 0.8, kernel = "Gaussian", penalty = "lasso")
beta.lasso = fit.lasso$coeff

#' ## Regularized conquer with scad penalty at tau = 0.8
fit.scad = conquer.reg(X, Y, lambda = 0.13, tau = 0.8, kernel = "Gaussian", penalty = "scad")
beta.scad = fit.scad$coeff
```

Index