Package 'qrjoint'

July 22, 2025

Version 2.0-9

Date 2023-03-30

Title Joint Estimation in Linear Quantile Regression

Author Surya Tokdar <surya.tokdar@duke.edu> and Erika Cunningham <erika.ball@duke.edu>

Maintainer Surya Tokdar <surya.tokdar@duke.edu>

Depends R (>= 2.6), stats, graphics, grDevices

Imports splines, coda, Matrix, kernlab, quantreg

Description
Joint estimation of quantile specific intercept and slope parameters in a linear regression setting.

License GPL-2

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-04-05 17:50:07 UTC

Contents

`chull.center` *Fast Interior Point Center of Multivariate Data*

Description

Calculates an interior point by averaging a small number of near-extreme points of the cloud.

Usage

```
chull.center(x, maxEPts = ncol(x) + 1, plot = FALSE)
```

Arguments

`x`	a matrix giving the data cloud.
`maxEPts`	integer giving the maximum number of (near)-extreme points to be used in averaging. Default is `ncol(x)+1`.
`plot`	logical indicating whether a pairwise scatter plot should be made

Details

Near extreme points are found in a space-filling manner by adding points with minimum residual conditional variance given points already included under a smooth GP specification. See Yang and Tokdar (2015), Section B.1. for more details.

Value

Returns an interior point of the data cloud. The positions of the near extreme points are returned as the attribute `"EPts"`.

Examples

```
p <- 9
n <- 200
u <- runif(n)
require(splines)
x <- bs(u, df = p)
chull.center(x, plot = TRUE)
```

coef.qde *Coefficient Extraction from qde Model Fit*

Description

Post process MCMC output from qde to create summaries of the quantile estimates.

Usage

```
## S3 method for class 'qde'
coef(object, burn.perc = 0.5, nmc = 200, reduce = TRUE, ...)
```

Arguments

object	a fitted model of the class qde.
burn.perc	a positive fraction indicating what fraction of the saved draws are to be discarded as burn-in
nmc	integer giving the number of samples, post burn-in, to be used in Monte Carlo averaging
reduce	logical indicating if the tail-expanded grid of tau values is to be reduced to the regular increment grid
...	not currently implemented

Value

Extracts posterior draws of intercept and slope functions from saved draws of model parameters. A plot may be obtained if plot = TRUE. A list is returned invisibly with three fields.

beta.samp	a matrix with nmc many columns and length(tau.grid) many rows.
beta.est	a 3-column matrix of median, 2.5th and 97.5th percentiles of the posterior distribution of β_0
parametric	a matrix with 3 columns giving the estimate (posterior median) and the lower and upper end points of the 95% posterior credible interval for γ_0, σ, and, ν.

See Also

qde and summary.qde for model fitting under qrjoint. Also see getBands for plotting credible bands for coefficients.

Examples

```
## Plasma data analysis
data(plasma)
Y <- plasma$BetaPlasma
Y <- Y + 0.1 * rnorm(length(Y)) ## remove atomicity
```

```
# model fitting with 50 posterior samples from 100 iterations (thin = 2)
fit.qde <- qde(Y, 50, 2)
betas <- coef(fit.qde)
signif(betas$parametric, 3)
```

`coef.qrjoint` *Regression Coefficient Extraction from qrjoint Model Fit*

Description

Post process MCMC output from `qrjoint` to create summaries of intercept and slope function estimates

Usage

```
## S3 method for class 'qrjoint'
coef(object, burn.perc = 0.5, nmc = 200, plot = FALSE, show.intercept = TRUE,
        reduce = TRUE, ...)
```

Arguments

`object`	a fitted model of the class `qrjoint`.
`burn.perc`	a positive fraction indicating what fraction of the saved draws are to be discarded as burn-in
`nmc`	integer giving the number of samples, post burn-in, to be used in Monte Carlo averaging
`plot`	logical indicating if plots are to be made
`show.intercept`	whether to plot the intercept curve when `plot = TRUE`
`reduce`	logical indicating if the tail-expanded grid of tau values is to be reduced to the regular increment grid
`...`	limited plotting parameters that are passed onto the call of `getBands`

Value

Extracts posterior draws of intercept and slope functions from saved draws of model parameters. A plot may be obtained if `plot = TRUE`. A list is returned invisibly with three fields.

`beta.samp`	a three-dimensional array of posterior samples of β_j. Its dimensions are `L` (length of tau.grid) x `p+1` (intercept + slopes) x `nmc` (retained posterior draws).
`beta.est`	a three-dimensional array containing posterior summaries (2.5th, 50th, and 97.5th percentiles) of β_j. Its dimensions are `L` (length of tau.grid) x `p+1` (intercept + slopes) x 3 (posterior summaries).
`parametric`	a matrix with 3 columns giving the estimate (posterior median) and the lower and upper end points of the 95% posterior credible interval for γ_0, γ, σ, and, ν.

See Also

qrjoint and summary.qrjoint for model fitting under qrjoint. Also see getBands for plotting credible bands for coefficients.

Examples

```
## Plasma data analysis

# recoding variables
data(plasma)
plasma$Sex <- as.factor(plasma$Sex)
plasma$SmokStat <- as.factor(plasma$SmokStat)
plasma$VitUse <- 3 - plasma$VitUse
plasma$VitUse <- as.factor(plasma$VitUse)

# model fitting with 50 posterior samples from 100 iterations (thin = 2)
fit.qrj <- qrjoint(BetaPlasma ~ Age + Sex + SmokStat + Quetelet + VitUse + Calories +
        Fat + Fiber + Alcohol + Cholesterol + BetaDiet, plasma, nsamp = 40, thin = 2)

## Not run:
betas <- coef(fit.qrj) ## no plots
betas <- coef(fit.qrj, plot = TRUE) ## estimates are plotted

## End(Not run)
```

---

getBands *Posterior Credible Bands*

Description

Calculate and display credible bands of a vector of parameters from a sample of draws. Most suitable when the vector represents a discretized version of a function.

Usage

```
getBands(b, col = 2, lwd = 1, plot = TRUE, add = FALSE,
x = seq(0,1,len=nrow(b)), remove.edges = TRUE, ...)
```

Arguments

b	a matrix of sampled draws of a vector, columns giving samples and rows giving elements of the vector
col	color of the median line and 95% bands, usual color codes could be used
lwd	line width for the median line
plot	logical indicating whether plots are to be drawn, default is TRUE
add	logical indicating whether plot is to be added to existing plot, default is FALSE

x	indexing the parameter coordinates. When b represents a (discretized) function, x can be taken as the function argument values. Needed when plot is to be created. Default sets it to a uniform grid of points over [0,1].
remove.edges	logical indicating whether the first and last entries of b are to be removed from plotting. This is helpful in qrjoint plots, where the two extremes could be Inf or nearly Inf.
...	limited number of plotting parameters

Value

returns median, 2.5th and 97.5th percentiles as a 3-column matrix.

See Also

coef.qrjoint

Examples

```
## toy example

x <- 2*pi*seq(0,1,.01)
fsamp <- replicate(100, rnorm(1,0,0.1) + rnorm(1,1,0.2) * cos(x))
getBands(fsamp)
getBands(fsamp, x = x, col = 3, remove.edges = FALSE, xlab = "x", ylab = "f", bty = "n")
getBands(fsamp, x = x, col = "darkgreen", remove.edges = FALSE, xlab = "x", ylab = "f")
```

---

plasma *Plasma Concentration of Beta-Carotene and Retinol*

Description

Plasma concentration levels of beta-carotene and retinol along with dietary intake and drug usage measurements for 315 patients who had an elective surgical procedure during a three-year period to biopsy or remove a lesion of the lung, colon, breast, skin, ovary or uterus that was found to be non-cancerous.

Usage

```
data(plasma)
```

Format

A data frame with 315 observations on the following 14 variables.

Age age (years)

Sex sex (1=Male, 2=Female)

SmokStat smoking status (1=Never, 2=Former, 3=Current)

`Quetelet` Quetlet index, aka, BMI (weight / height^2)

`VitUse` vitamin use (`0`=No, `1`=Yes, not often, `2`=Yes, fairly often)

`Calories` number of calories consumed per day

`Fat` grams of fat consumed per day

`Fiber` grams of fiber consumed per day

`Alcohol` number of alcoholic drinks consumed per week

`Cholesterol` cholesterol consumed (mg per day)

`BetaDiet` dietary beta-carotene consumed (mcg per day)

`RetDiet` dietary retinol consumed (mcg per day)

`BetaPlasma` plasma beta-carotene concentration (ng/ml)

`RetPlasma` plasma retinol concentration (ng/ml)

Details

Dietary intakes are self-reported. Results from analyzing this data should be used with caution!

Source

Statlib database

References

Nierenberg, D. W., T. A. Stukel, J. A. Baron, B. J. Dain, and E. R. Greenberg (1989). Determinants of plasma levels of beta-carotene and retinol. *American Journal of Epidemiology*, 130(3), 511–521.

Examples

```
data(plasma)
```

---

predict.qde *Posterior predictive summary for quantile-based density estimation*

Description

Extract posterior predictive density estimate for qde

Usage

```
## S3 method for class 'qde'
predict(object, burn.perc = 0.5, nmc = 200, yRange = range(object$y), yLength = 401, ...)
```

Arguments

object	a fitted model of the class 'qde'.
burn.perc	a positive fraction indicating what fraction of the saved draws are to be discarded as burn-in
nmc	integer giving the number of samples, post burn-in, to be used in Monte Carlo averaging
yRange	range of values over which posterior predictive density is to be evaluated
yLength	number of grid points spanning yRange for posterior predictive density evaluation
...	currently no additional arguments are allowed

Value

Returns a list with three items:

y	vector giving the grid over which the posterior predictive density is evaluated.
fsamp	a matrix with yLength many rows and nmc many columns. Each column corresponds to a draw of the response density from the posterior predictive.
fest	summary of the posterior predictive density given by point-wise median, 2.5th and 97.5th percentiles.

See Also

qde and summary.qde.

Examples

```
# Plasma data analysis

data(plasma)
Y <- plasma$BetaPlasma
Y <- Y + 0.1 * rnorm(length(Y)) ## remove atomicity

# model fitting with 50 posterior samples from 100 iterations (thin = 2)
fit.qde <- qde(Y, 50, 2)
pred <- predict(fit.qde)
hist(Y, freq = FALSE, col = "gray", border = "white", ylim = c(0, max(pred$fest)))
matplot(pred$y, pred$fest, type="l", col=1, lty=c(2,1,2), ylab="Density", xlab="y")
```

predict.qrjoint *Posterior predictive summary for quantile estimation*

Description

Extract quantile functions for qrjoint

Usage

```
## S3 method for class 'qrjoint'
predict(object, newdata=NULL, summarize=TRUE, burn.perc = 0.5, nmc = 200, ...)
```

Arguments

object	a fitted model of the class 'qrjoint'.
newdata	an optional data frame containing variables on which to predict. If omitted, the fitted data are used.
summarize	a logical indicating whether the quantile functions should be summarized across posterior draws into a single estimate (TRUE) or be left as individual samples (FALSE)
burn.perc	a positive fraction indicating what fraction of the saved draws are to be discarded as burn-in
nmc	integer giving the number of samples, post burn-in, to be used in Monte Carlo averaging
...	currently no additional arguments are allowed

Value

Either returns a matrix of posterior quantile-function estimates if summarize=TRUE. Dimensions are n (number of rows in predicted data) x L (length of regularized tau.grid); or a three dimensional array of posterior quantile-function samples if summarize=FALSE. Dimensions are n (number of rows in predicted data) x L (length of regularized tau.grid) x nmc (retained posterior draws).

Note

When supplying newdata, new covariate values should lie within the convex hull of the original fit's covariate space; otherwise, it is possible that extrapolated quantile functions will not obey the quantile monotonicity constraint. For information on expanding the convex hull of the original fit see Detail section of qrjoint.

Additional functionality is planned in future release to provide density function estimates given a set of input covariates.

See Also

qrjoint and summary.qrjoint.

Examples

```
## Plasma data analysis

# recoding variables
data(plasma)
plasma$Sex <- as.factor(plasma$Sex)
plasma$SmokStat <- as.factor(plasma$SmokStat)
plasma$VitUse <- 3 - plasma$VitUse
plasma$VitUse <- as.factor(plasma$VitUse)

# Model fitting with 40 posterior samples from 80 iterations (thin = 2) is for
# illustration only. For practical model fitting, increase iterations,
# e.g. nsamp = 500, thin = 20
## Not run:
fit.qrj <- qrjoint(BetaPlasma ~ Age + Sex + SmokStat + Quetelet + VitUse + Calories +
        Fat + Fiber + Alcohol + Cholesterol + BetaDiet, plasma, nsamp = 40, thin = 2)

quants <- predict(fit.qrj)
matplot(fit.qrj$tau.g[fit.qrj$reg.ix], t(quants), type="l", xlab="p",
ylab="Quantile Function", col="lightgray", lty=1)

## End(Not run)
```

---

qde *Quantiles based Density Estimation*

Description

Provides a semiparametric estimation of the quantiles for independented univariate data with possible right censoring. This is same as estimating the intercept function within a joint linear quantile regression model with no predictors.

Usage

```
qde(y, nsamp = 1e3, thin = 10, cens = NULL,
    wt = NULL, incr = 0.01, par = "prior", nknots = 6,
    hyper = list(sig = c(.1,.1), lam = c(6,4), kap = c(0.1,0.1,1)),
    prox.range = c(.2,.95), acpt.target = 0.15, ref.size = 3,
    blocking = "single", temp = 1, expo = 2, blocks.mu, blocks.S,
    fix.nu = FALSE, fbase = c("t", "logistic", "unif"), verbose = TRUE)

## S3 method for class 'qde'
update(object, nadd, append = TRUE, ...)
```

Arguments

y	numeric vector of response data.
nsamp	number of posterior samples to be saved; defaults to 1000.

`thin` thinning rate for the Markov chain sampler – one posterior sample is saved per `thin` iterations. Defaults to 10. The Markov chain sampler runs for a total of `nsamp * thin` many iterations.

`cens` censoring status of response. Must be a vector of length length(y), with 0 indicating no censoring, 1 indicating right censoring, and 2 indicating left censoring. If not supplied, defaults to all zeros.

`wt` weights attached to the observation units, expected to be non-negative numbers, and defaults to a vector of ones if not otherwise supplied.

`incr` tau grid increment. Defaults to 0.01.

`par` character string indicating how the sampler is to be initialized. Only two options are currently supported: "prior" to initialize at a random draw from the prior; "RQ" to initialize at a model space approximation of the estimates from `rq`.

`nknots` number of knots to be used for low rank approximation of the Gaussian process priors. Defaults to 6.

`hyper` hyperparameters of the prior distribution. Must be a list with some of all of the following fields: `sig`: a two vector giving the parameters of the inverse-gamma distribution on sigma-square that is used when `shrink`=TRUE, `lam`: a two vector giving the parameters of the beta distribution on proximity = $\exp(-0.01 * \lambda^2)$, and `kap`: a vector to be coerced into a `3 * nkap` matrix, with `nkap` being the number of components in the mixture of gamma prior on `kappa`, and each column of the matrix gives the shape, rate and mixing weight of a component.

`prox.range` for specifying the range of length-scale parameter of the Gaussian process prior.

`acpt.target` target acceptance rate of the adaptive Metropolis sampler; defaults to 0.15

`ref.size` adaptation rate of the adaptive Metropolis sampler. The proposal density is updated once every `ref.size` iterations. Could be a single number or a vector of length same as the number of blocks.

`blocking` type of blocking to be applied. Either a character string specifying one to be chosen from the supplied menu (see Details), or a list giving user specified blocks. In the latter case, each element of the list is a logical vector of length equal to the total number of model parameters, which equals `nknots+3` indicating which model parameters belong to the block.

`temp` temperature of the log-likelihood function. The log-likelihood function is raised to the power of `temp`. Defaults to 1.

`expo` the exponent to be used in the covariance kernel of the Gaussian process priors. Defaults to 2, giving the standard squared-exponential covariance kernel.

`blocks.mu` initial block specific means in the form of a list. If left unspecified then will be automatically generated as a list of vectors of zeros of appropriate lengths matching the corresponding block sizes.

`blocks.S` initial block specific covariance matrices in the form of a list. If left unspecified then will be automatically generated as a list of identity matrices of appropriate dimensions matching the corresponding block sizes. When `blocking` is chosen as one of the menu items of the form "std*", known prior covariance information and estimated variance matrices from `rq` are used.

fix.nu	either the logical FALSE indicating that nu should be learned, or a positive real number giving the fixed value of nu, which is then excluded from MCMC updates
fbase	either "t" (default), "logistic" or "unif" to indicate what base distribution is to be used.
verbose	logical indicating whether MCMC progress should be printed, defaults to TRUE
object	a fitted model of the class 'qde'.
nadd	number of additional MCMC samples.
append	logical indicating whether new samples should be appended to old ones. If FALSE then old samples are discarded.
...	no additional arguments are allowed

Details

The model assumes the quantile function of the data is given by: `Q(t) = gamma_0 + sigma * (Qb(zeta(t)|nu) - Qb(zeta(0,5)|nu))`, `Qb(.|nu)` is a parametric quantile function with unknown parameter `nu`, `gamma_0` is the unknown median, `sigma` is an unknown saling factor, and, `zeta` is an unknown distortion of the unit interval. The distortion `zeta` is modeled nonparametrically through a logistic Gaussian process prior, other parameters are given diffuse priors.

In running the MCMC, the following menu choices are available for blocking the parameter vector. For this special case `p = ncol(X) = 0`, some of the menu choices are actually the same, in particular, "std0" is same as "single", "std1" is same as "single2", and, "std2" is same as "single3".

`"single"`: a single block containing all parameters

`"single2"`: one block containing all parameters and an additional block containing only (gamma[0], gamma, sigma, nu)

`"single3"`: like `"single2"`, but the second block is split into two further blocks, one with (γ_0, γ), the other with (σ, ν)

`"std0"`: Same as `"single"`.

`"std1"`: Same as `"single2"`.

`"std2"`: Same as `"single3"`.

`"std3"`: total 3 blocks. First block is W_{*0}, last two are (γ_0, γ) and (σ, ν)

`"std4"`: total 3 blocks. First block is (W_{*0}, γ_0), last two are (γ_0, γ) and (σ, ν)

`"std5"`: total 4 blocks. First three are same as `"std4"` and one additional block containing all parameters.

Value

`qde(y, ...)` returns a 'qde' class object to be used by `coef` and `summary`.

Returned object is a list containing the following variables.

par	latest draw of the parameter vector
y	response vector
cens	censoring status vector

wt	vector of observation weights
hyper	completed list of hyper-parameters
dim	model dimension vector of the form c(n, p, length of tau grid, position of τ_0 on the grid, nknots, length of lambda grid, nkap, total number of MCMC iterations, thin, nsamp)
gridmats	details of covariance matrix factors etc, intended for internal use.
tau.g	the tau grid
muV	list of means for parameter blocks
SV	list of covariance matrices for parameter blocks
blocks	list of blocks
blocks.size	vector of block lengths
dmcmcpar	numeric vector containing details of adaptive MCMC runs, equals c(temp, decay rate of adaptation, vector of target acceptance rates for the blocks, vector of increment scales used in adaptation). Intended strictly for internal use.
imcmcpar	numeric vector containing details of adaptive MCMC runs, equals c(number of parameter blocks, ref.size, indicator on whether details are to be printed during MCMC progress, rate of details printing, a vector of counters needed for printing). Intended strictly for internal use.
parsamp	a long vector containing the parameter draws. Could be coerced into a matrix of dim npar * nsamp. Intended primarily for use by summary and coef.
acptsamp	a long vector containing rates of acceptance statistics for parameter blocks. Could be coerced into a matrix of dim nblocks * nsamp. Not very informative, because thinning times and adaptation times may not be exactly synced.
lpsamp	vector of log posterior values for the saved MCMC draws.
fbase.choice	integer 1 for "t", 2 for "logistic" and 3 for "unif" base.
prox	vector of proximity (exp(-0.01*lambda^2)) grid values
reg.ix	positions of the regular tau grid on the expanded tail-appended grid
runtime	run time of the MCMC

References

Yang, Y. and Tokdar, S.T., 2017. Joint estimation of quantile planes over arbitrary predictor spaces. Journal of the American Statistical Association, 112(519), pp.1107-1120.

See Also

summary.qde, coef.qde and predict.qde. Also see qrjoint for regression model fitting in presence of covariates.

Examples

```
## Plasma data analysis

data(plasma)
Y <- plasma$BetaPlasma

# model fitting with 100 posterior samples from 200 iterations (thin = 2)
# this is of course for illustration, for practical model fitting you
# ought to try at least something like nsamp = 500, thin = 20
fit.qde <- qde(Y, nsamp = 100, thin = 2)
summary(fit.qde, more = TRUE)
pred <- predict(fit.qde)
hist(Y, freq = FALSE, col = "gray", border = "white", ylim = c(0, max(pred$fest)))
lines(pred$y, pred$fest[,2])
lines(pred$y, pred$fest[,1], lty = 2)
lines(pred$y, pred$fest[,3], lty = 2)
```

`qrjoint` *Joint Estimation of Linear Quantile Planes*

Description

Estimate intercept and slope functions within a joint linear regression model of the quantiles, with possible right or left censoring of the response.

Usage

```
qrjoint(formula, data, nsamp = 1e3, thin = 10, cens = NULL,
    wt = NULL, incr = 0.01, par = "prior", nknots = 6,
    hyper = list(sig = c(.1,.1), lam = c(6,4), kap = c(.1,.1,1)),
    shrink = FALSE, prox.range = c(.2,.95), acpt.target = 0.15,
    ref.size = 3, blocking = "std5", temp = 1, expo = 2,
   blocks.mu, blocks.S, fix.nu=FALSE, fbase = c("t", "logistic", "unif"), verbose = TRUE)

## S3 method for class 'qrjoint'
update(object, nadd, append = TRUE, ...)
```

Arguments

`formula`	an object of class "formula": a symbolic description of the model to be fitted. It must include at least one predictor.
`data`	a data frame containing variables in the model.
`nsamp`	number of posterior samples to be saved; defaults to 1000.
`thin`	thinning rate for the Markov chain sampler – one posterior sample is saved per `thin` iterations. Defaults to 10. The Markov chain sampler runs for a total of `nsamp` $*$ `thin` many iterations.

`cens` censoring status of response. Must be a vector of length NROW(data), with 0 indicating no censoring, 1 indicating right censoring, and 2 indicating left censoring. If not supplied, defaults to all zeros.

`wt` weights attached to the observation units, expected to be non-negative numbers, and defaults to a vector of ones if not otherwise supplied.

`incr` tau grid increment. Defaults to 0.01.

`par` character string indicating how the sampler is to be initialized. Three options are currently supported: "prior" to initialize at a random draw from the prior; "RQ" to initialize at a model space approximation of the estimates from `rq`; and, "noX" to initialize at a model with all slope functions being equal to zero, and the intercept function obtained by fitting `qde` to the response data alone.

`nknots` number of knots to be used for low rank approximation of the Gaussian process priors. Defaults to 6.

`hyper` hyperparameters of the prior distribution. Must be a list with some of all of the following fields: `sig`: a two vector giving the parameters of the inverse-gamma distribution on sigma-square that is used when `shrink`=TRUE, `lam`: a two vector giving the parameters of the beta distribution on proximity = $\exp(-0.01 * \lambda^2)$, and `kap`: a vector to be coerced into a `3 * nkap` matrix, with `nkap` being the number of components in the mixture of gamma prior on `kappa`, and each column of the matrix gives the shape, rate and mixing weight of a component.

`shrink` for applying shrinkage to gamma[0] and gamma. Defaults to FALSE, in which case a right Haar prior is used on (gamma[0], gamma, sigma2). If TRUE then a horseshoe type prior is used.

`prox.range` for specifying the range of length-scale parameter of the Gaussian process prior.

`acpt.target` target acceptance rate of the adaptive Metropolis sampler; defaults to 0.15

`ref.size` adaptation rate of the adaptive Metropolis sampler. The proposal density is updated once every `ref.size` iterations. Could be a single number or a vector of length same as the number of blocks.

`blocking` type of blocking to be applied. Either a character string specifying one to be chosen from the supplied menu (see Details), or a list giving user specified blocks. In the latter case, each element of the list is a logical vector of length equal to the total number of model parameters, which equals `(nknots+1)*(ncol(X)+1) + 2` indicating which model parameters belong to the block.

`temp` temperature of the log-likelihood function. The log-likelihood function is raised to the power of `temp`. Defaults to 1.

`expo` the exponent to be used in the covariance kernel of the Gaussian process priors. Defaults to 2, giving the standard squared-exponential covariance kernel.

`blocks.mu` initial block specific means in the form of a list. If left unspecified then will be automatically generated as a list of vectors of zeros of appropriate lengths matching the corresponding block sizes.

`blocks.S` initial block specific covariance matrices in the form of a list. If left unspecified then will be automatically generated as a list of identity matrices of appropriate dimensions matching the corresponding block sizes. When `blocking` is chosen as one of the menu items of the form "std*", known prior covariance information and estimated variance matrices from `rq` are used.

fix.nu	either the logical FALSE indicating that nu should be learned, or a positive real number giving the fixed value of nu, which is then excluded from MCMC updates
fbase	either "t" (default), "logistic" or "unif" to indicate what base distribution is to be used.
verbose	logical indicating whether MCMC progress should be printed, defaults to TRUE
object	a fitted model of the class 'qrjoint'.
nadd	number of additional MCMC samples.
append	logical indicating whether new samples should be appended to old ones. If FALSE then old samples are discarded.
...	no additional arguments are allowed

Details

A formula has an implied intercept term. This model requires that the intercept term be included; therefore, it cannot be explicitely removed via (y ~ 0 + x) or (y ~ -1 + x) constructs.

The model assumes each conditional quantile of the response is a hyper-plane. The intercept and slope functions (as functons of the quantile level) are estimated under the constraint that the resulting quantile planes are non-crossing over some closed, convex predictor domain. The domain is equated, by default, to the convex hull of the observed predictor vectors. The input argument `wt` provides more flexibility in the domain specification. All observation units are used in calculating the convex hull, but only those with non-zero weights are used in the likelihood evaluation. By including pseudo-points with zero weight (e.g. covariates from a test dataframe), the boundaries of the predictor domain can be expanded.

In running the MCMC, the following menu choices are available for blocking the parameter vector. Below, `p = ncol(X)`.

`"single"`: a single block containing all parameters

`"single2"`: one block containing all parameters and an additional block containing only (gamma[0], gamma, sigma, nu)

`"single3"`: like `"single2"`, but the second block is split into two further blocks, one with (γ_0, γ), the other with (σ, ν)

`"std0"`: `p+1` blocks, $(j+1)$-th contains $(W_{*j}, \gamma_j, \sigma, \nu)$, $j = 0, \ldots, p$.

`"std1"`: total `p+2` blocks. First `p+1` blocks same as `"std0"` and one additional block of $(\gamma_0, \gamma, \sigma, \nu)$.

`"std2"`: total `p+3` blocks. First `p+1` blocks same as `"std0"` and two additional blocks of (γ_0, γ) and (σ, ν)

`"std3"`: total `p+3` blocks. First `p+1` blocks are W_{*j}, $j = 0, \ldots, p$, last two are (γ_0, γ) and (σ, ν)

`"std4"`: total `p+3` blocks. First `p+1` blocks are (W_{*j}, γ_j), $j = 0, \ldots, p$, last two are (γ_0, γ) and (σ, ν)

`"std5"`: total `p+4` blocks. First `p+3` are same as `"std4"` and one additional block containing all parameters.

Value

`qrjoint(x, y, ...)` returns a 'qrjoint' class object to be used by `update.qrjoint`, `coef.qrjoint` and `summary.qrjoint`.

`update(object,...)` runs additional Markov chain iterations and appends thinned draws to an existing 'qrjoint' object `object`. All relevant details are inherited from `object`.

Returned object is a list containing the following variables.

`par`	latest draw of the parameter vector
`x`	scaled and centered design matrix
`y`	response vector
`cens`	censoring status vector, 0=uncensored, 1=right censored, 2=left censored
`wt`	vector of observation weights
`shrink`	shrinkage indicator
`hyper`	completed list of hyper-parameters
`dim`	model dimension vector of the form c(n, p, length of tau grid, position of τ_0 on the grid, nknots, length of lambda grid, nkap, total number of MCMC iterations, thin, nsamp)
`gridmats`	details of covariance matrix factors etc, intended for internal use.
`tau.g`	the tau grid
`muV`	list of means for parameter blocks
`SV`	list of covariance matrices for parameter blocks
`blocks`	list of blocks
`blocks.size`	vector of block lengths
`dmcmcpar`	numeric vector containing details of adaptive MCMC runs, equals c(temp, decay rate of adaptation, vector of target acceptance rates for the blocks, vector of increment scales used in adaptation). Intended strictly for internal use.
`imcmcpar`	numeric vector containing details of adaptive MCMC runs, equals c(number of parameter blocks, ref.size, indicator on whether details are to be printed during MCMC progress, rate of details printing, a vector of counters needed for printing). Intended strictly for internal use.
`parsamp`	a long vector containing the parameter draws. Could be coerced into a matrix of dim `npar * nsamp`. Intended primarily for use by `summary.qrjoint` and `coef.qrjoint`.
`acptsamp`	a long vector containing rates of acceptance statistics for parameter blocks. Could be coerced into a matrix of dim `nblocks * nsamp`. Not very informative, because thinning times and adaptation times may not be exactly synced.
`lpsamp`	vector of log posterior values for the saved MCMC draws.
`fbase.choice`	integer 1 for "t", 2 for "logistic" and 3 "unif" base.
`prox`	vector of proximity (exp(-0.01*lambda^2)) grid values
`reg.ix`	positions of the regular tau grid on the expanded tail-appended grid
`runtime`	run time of the MCMC
`call`	original model call
`terms`	terms included in model frame

References

Yang, Y. and Tokdar, S.T., 2017. Joint estimation of quantile planes over arbitrary predictor spaces. Journal of the American Statistical Association, 112(519), pp.1107-1120.

See Also

summary.qrjoint and coef.qrjoint.

Examples

```
## Plasma data analysis

# recoding variables
data(plasma)
plasma$Sex <- as.factor(plasma$Sex)
plasma$SmokStat <- as.factor(plasma$SmokStat)
plasma$VitUse <- 3 - plasma$VitUse
plasma$VitUse <- as.factor(plasma$VitUse)

# Model fitting with 40 posterior samples from 80 iterations (thin = 2) is for
# illustration only. For practical model fitting, increase iterations,
# e.g. nsamp = 500, thin = 20
fit.qrj <- qrjoint(BetaPlasma ~ Age + Sex + SmokStat + Quetelet + VitUse + Calories +
        Fat + Fiber + Alcohol + Cholesterol + BetaDiet, plasma, nsamp = 40, thin = 2)
summary(fit.qrj, more = TRUE)

## Not run:
# additional MCMC runs to get 10 more samples (20 additional iterations)
fit.qrj <- update(fit.qrj, 10)
summary(fit.qrj, more = TRUE)

## End(Not run)

## Not run:
### UIS data analysis (with right censoring)
data(uis)
uis.qrj <- qrjoint(Y ~ TREAT + NDT + IV3 + BECK + FRAC +
                      RACE + AGE + SITE , data=uis, cens = (1 - uis$CENSOR),
                    nsamp = 50, thin = 2, fix.nu = 1e5)
summary(uis.qrj, more = TRUE)

betas <- coef(uis.qrj, plot = TRUE, col = "darkgreen")
tau.grid <- uis.qrj$tau.g[uis.qrj$reg.ix]
L <- length(tau.grid)
beta.samp <- betas$beta.samp

# survival curve estimation for k randomly chosen subjects
n <- nrow(uis)
k <- 9
ix.sel <- sort(sample(n, k))
Qsel.gp <- predict(uis.qrj, newdata=uis[ix.sel,], summarize=FALSE)
```

```
colRGB <- col2rgb("darkgreen")/255
colTrans <- rgb(colRGB[1], colRGB[2], colRGB[3], alpha = 0.05)
par(mfrow = c(3,3), mar = c(4,3,2,1) + .1)
for(i in 1:k){
  plot(exp(apply(Qsel.gp[i,,],1,mean)), 1 - tau.grid, ty = "n", ann = FALSE,
       bty = "n", xlim = exp(c(2, 8)), ylim = c(0,1), lty = 2, log = "x")
  for(j in 1:dim(beta.samp)[3])
      lines(exp(Qsel.gp[i,,j]), 1 - tau.grid, col = colTrans, lwd = 1)
  title(xlab = "Return time (days)", ylab = "Survival function", line = 2)
  title(main = bquote(Obs.Id == .(ix.sel[i])))
  grid()
}

## End(Not run)
```

`redmaple` *Basal Areas of Red Maple Trees*

Description

Site-level basal areas of red maple trees (*Acer rubrum*) at 608 unmanaged and forested sites in Connecticut, Massachusetts, and Rhode Island. Data are aggregated from the Forest Inventory Analysis (FIA) of the US Forest Service. Geographical regions of the sites are added using Enviromental Protection Agency (EPA) shapefiles.

Usage

```
data(redmaple)
```

Format

A data frame with 608 observations on the following variables for each site:

`plotID` Unique identifier

`elev` Elevation, measured in feet

`slope` Slope, measured in degrees

`aspect` Aspect, measured in degrees proceeding from North clockwise around a compass. For sites with zero or near-zero slopes, aspect is recorded as 0. North is recorded as 360.

`Region` EPA Level-III geographical region

`region` EPA Level-III geographical region, shortened

`State` State

`baRedMaple` Total basal area of red maple trees, measured in square feet per acre

Details

This three-state subset from the FIA is intended to illustrate the capabilities of the qrjoint package in flexibly modeling excess-boundary zeros, using its censoring option. This subset of variables should not be construed as a comprehensive list of factors influencing red maple basal area growth. All sites in the sample are of equivalent area.

Source

Forest Inventory and Analysis Database, St. Paul, MN: U.S. Department of Agriculture, Forest Service, Northern Research Station. https://apps.fs.usda.gov/fia/datamart/datamart.html

Examples

```
data(redmaple)
```

`summary.qde` *Summary Method for Quantile based Density Estimation*

Description

Summarize model fit for qde

Usage

```
## S3 method for class 'qde'
summary(object, ntrace = 1000, burn.perc = 0.5,
        plot.dev = TRUE, more.details = FALSE, ...)
```

Arguments

`object`	a fitted model of the class 'qde'.
`ntrace`	number of draws to be included in trace plots
`burn.perc`	fraction of MCMC draws to be discarded as burn-in.
`plot.dev`	logical indicator of whether to show trace plot of deviance
`more.details`	logical indicating whether other details from MCMC are to be plotted
`...`	a limited number of plotting controls that are passed onto the deviance plot

Value

Displays the trace of the deviance statistic. More details include trace plots of of the proximity parameter of each GP, a plot of Geweke p-values for (from geweke.diag) convergence of each model parameter and an image plot of parameter correlation. Also prints two versions of Watanabe AIC.

The following quantities are returned invisibly.

`deviance`	vector deviance statistic of the samples parameter draws
`pg`	a matrix with `nsamp` number of columns, each columns could be coerced into a matrix of dimension `ngrid * (p+1)`, where the columns gives the conditional posterior weights on the lambda grid values for the corresponding GP function.
`prox`	posterior draws of proximity in the form of a `(p+1)*nsamp` matrix.
`ll`	a matrix of `n*nsamp` containing observation-level log-likelihood contributions. Used to calculate *waic*, and could be used for other AIC calculations.
`ql`	a matrix of `n*nsamp` containing observation-level estimated quantile levels.
`waic`	Two versions of Watanabe AIC from Gelman, Hwang and Vehtari (2014).

References

Gelman, A., Hwang, J., and Vehtari, A. (2014). Understanding predictive information criterion for Bayesian models. *Stat Comput*, 24, 997-1016.

See Also

qrjoint and coef.qrjoint.

Examples

```
# Plasma data analysis

data(plasma)
Y <- plasma$BetaPlasma
Y <- Y + 0.1 * rnorm(length(Y)) ## remove atomicity

# model fitting with 50 posterior samples from 100 iterations (thin = 2)
fit.qde <- qde(Y, 50, 2)
summary(fit.qde, more = TRUE)
```

---

summary.qrjoint *Summary Method for qrjoint Model Fit*

Description

Summarize model fit, including MCMC details, for qrjoint.

Usage

```
## S3 method for class 'qrjoint'
summary(object, ntrace = 1000, burn.perc = 0.5,
        plot.dev = TRUE, more.details = FALSE, ...)
```

Arguments

object	a fitted model of the class 'qrjoint'.
ntrace	number of draws to be included in trace plots
burn.perc	fraction of MCMC draws to be discarded as burn-in.
plot.dev	logical indicator of whether to show trace plot of deviance
more.details	logical indicating whether other details from MCMC are to be plotted
...	a limited number of plotting controls that are passed onto the deviance plot

Value

Displays the trace of the deviance statistic. More details include trace plots of of the proximity parameter of each GP, a plot of Geweke p-values for (from geweke.diag) convergence of each model parameter and an image plot of parameter correlation. Also prints two versions of Watanabe AIC.

The following quantities are returned invisibly.

deviance	vector deviance statistic of the samples parameter draws
pg	a matrix with nsamp number of columns, each columns could be coerced into a matrix of dimension ngrid * (p+1), where the columns gives the conditional posterior weights on the lambda grid values for the corresponding GP function.
prox	posterior draws of proximity in the form of a (p+1)*nsamp matrix.
ll	a matrix of n*nsamp containing observation-level log-likelihood contributions. Used to calculate *waic*, and could be used for other AIC calculations.
ql	a matrix of n*nsamp containing observation-level estimated quantile levels (i.e. t such that $y = Q(t\|x)$) at each posterior draw. These may be used in lieu of residuals to assess model fit and assumption of linearity.
waic	Two versions of Watanabe AIC from Gelman, Hwang and Vehtari (2014).

References

Gelman, A., Hwang, J., and Vehtari, A. (2014). Understanding predictive information criterion for Bayesian models. *Stat Comput*, 24, 997-1016.

See Also

qrjoint and coef.qrjoint.

Examples

```
# Plasma data analysis

# recoding variables
data(plasma)
plasma$Sex <- as.factor(plasma$Sex)
plasma$SmokStat <- as.factor(plasma$SmokStat)
plasma$VitUse <- 3 - plasma$VitUse
plasma$VitUse <- as.factor(plasma$VitUse)

# Model fitting with 40 posterior samples from 80 iterations (thin = 2) is for
# illustration only. For practical model fitting, increase iterations,
# e.g. nsamp = 500, thin = 20
fit.qrj <- qrjoint(BetaPlasma ~ Age + Sex + SmokStat + Quetelet + VitUse + Calories +
        Fat + Fiber + Alcohol + Cholesterol + BetaDiet, plasma, nsamp = 40, thin = 2)
summ <- summary(fit.qrj, more = TRUE)

## Not run:
# Visually assess uniformity of quantile levels with histogram and qqplot
# Notes: Can assess across all MCMC draws (as below) or for single iteration;
```

```
# adjustments to quantile levels will be needed for censored observations
hist(summ$ql, breaks=40, freq=F)
curve(dunif(x),add=T)
qqplot(summ$ql, qunif(ppoints(length(summ$ql))),xlab="actual", ylab="theoretical")
abline(0,1)

# Visually assess linearity assumption using quantile levels
# Notes: Can assess across all MCMC draws or for single iteration (as below)

# Loess gives visual of center of quantile levels across covariate;
# trend line should be near 0.5
library(ggplot2)
use <- sample(1:ncol(summ$ql),1)
plasma$qlsamp <- summ$ql[,use]
ggplot(data=plasma, aes(x=Age, y=qlsamp)) + geom_point() + geom_smooth(se=F,
method="loess")

# Violin plot allows for assessment of entire distribution across covariate;
# densities within decile bins should be blocky-uniform
cut_dec <- function(x) factor(cut(x, quantile(x,0:10/10),inc=TRUE),labels=1:10)
ggplot(data=plasma, aes(x=cut_dec(Age), y=qlsamp)) + geom_violin() +
xlab("Age Decile Bins")

## End(Not run)
```

`waic` *Watanabe Information Criterion*

Description

Calculates two versions of the Watanabe information criteria from MCMC draws.

Usage

```
waic(logliks, print = TRUE)
```

Arguments

`logliks`	a matrix of observation level log-likelihood values, the columns are MCMC iterations and the rows are observations in the data
`print`	logical whether to print the results

Value

Returns the two version of the WAIC

References

Gelman, A., Hwang, J., and Vehtari, A. (2014). Understanding predictive information criterion for Bayesian models. *Stat Comput*, 24, 997-1016.

See Also

summary.qrjoint

Examples

```
# Plasma data analysis

# recoding variables
data(plasma)
plasma$Sex <- as.factor(plasma$Sex)
plasma$SmokStat <- as.factor(plasma$SmokStat)
plasma$VitUse <- 3 - plasma$VitUse
plasma$VitUse <- as.factor(plasma$VitUse)

# Model fitting with 40 posterior samples from 80 iterations (thin = 2) is for
# illustration only. For practical model fitting, increase iterations,
# e.g. nsamp = 500, thin = 20
fit.qrj <- qrjoint(BetaPlasma ~ Age + Sex + SmokStat + Quetelet + VitUse + Calories +
        Fat + Fiber + Alcohol + Cholesterol + BetaDiet, plasma, nsamp = 40, thin = 2)
summary(fit.qrj, more = TRUE)

# the call to summary already shows the waic for the fitted model, it also returns
# the observation level log-likelihood vales. To calculate waic from last 20 draws
# we can use:

## Not run:
summary(fit.qrj, more = TRUE)
ll <- sm$ll
waic(ll[,21:40])

## End(Not run)
```

Index