Package 'tauBayesW'

September 12, 2025

Title Bayesian Weighted Quantile Regression with EM and MCMC Algorithm

Version 0.1.0

Description Implements Bayesian quantile regression approaches using the EM algorithm and several MCMC methods with observation weights for complex survey designs. Includes fast C++ implementations using 'Rcpp', 'RcppArmadillo', and 'RcppEigen'.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

LinkingTo Rcpp,
RcppArmadillo,
RcppEigen

Imports Rcpp,
stats,
graphics,
methods,
plotly,
geometry,
pracma,
ggplot2,
rlang,
posterior

Suggests MASS,
knitr,
rmarkdown,
grDevices,
testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

SystemRequirements BLAS, LAPACK

URL https://github.com/torodriguezt/tauBayesW

BugReports https://github.com/torodriguezt/tauBayesW/issues

Contents

`bqr.svy` *Bayesian Weighted Quantile Regression (Survey Design)*

Description

Fits a Bayesian quantile regression model with survey weights using one of three MCMC kernels implemented in C++:

- `.MCMC_BWQR_AL` – Asymmetric Laplace Distribution
- `.MCMC_BWQR_SL` – Score likelihood
- `.MCMC_BWQR_AP` – Approximate likelihood

One or more quantiles can be estimated, depending on the input.

Usage

```
bqr.svy(
  formula,
  weights = NULL,
  data = NULL,
  quantile = 0.5,
  method = c("ald", "score", "approximate"),
  prior = NULL,
  niter = 50000,
  burnin = 10000,
  thin = 1,
  print_progress = 1000,
  ...
)
```

Arguments

`formula`	A `formula` specifying the model.
`weights`	Optional survey weights (numeric vector or one-sided formula). Weights are passed directly to the underlying C++ algorithms without any preprocessing like scaling.
`data`	Optional `data.frame` containing the variables used in the model.
`quantile`	Numeric scalar or vector in (0, 1): target quantile(s) τ. Duplicates are automatically removed.

method	One of "ald", "score", "approximate". Default is "ald" (Asymmetric Laplace Distribution).
prior	Prior specification. Can be: • A bqr_prior object from prior • A list with components b0, B0, and optionally c0, C0 • NULL (uses default vague priors) For "ald": uses b0, B0, c0, C0. For "score" and "approximate": uses b0, B0 only. **Tip:** Use prior() for a simpler unified interface.
niter	Integer. Number of MCMC iterations.
burnin	Integer. Number of burn-in iterations.
thin	Integer. Thinning interval.
print_progress	Integer. Print progress every print_progress iterations. Set to 0 to disable progress printing. Default is 1000.
...	Additional arguments passed to underlying functions (reserved for future use).

Details

Survey weights are handled differently by each method:

- "ald" and "score": weights are normalized (divided by their mean).
- "approximate": weights are used as provided (raw weights).

Prior Specification:

The prior can be specified in several ways:

1. Using prior (recommended).
2. As a list with b0, B0, and optionally c0, C0.
3. As NULL, in which case vague priors are used.

Multiple quantiles can be fitted in a single call. The returned object adapts its class accordingly ("bwqr_fit" for one quantile, "bwqr_fit_multi" for several).

Value

An object of class "bqr.svy", containing:

beta	Posterior mean estimates of regression coefficients.
draws	Posterior draws from the MCMC sampler.
accept_rate	Average acceptance rate (if available).
quantile	The quantile(s) fitted.
prior	Prior specification used.
formula, terms, model	Model specification details.
runtime	Elapsed runtime in seconds.

Examples

```
# Generate example data
set.seed(123)
n <- 100
x1 <- rnorm(n)
x2 <- runif(n, -1, 1)
y <- 2 + 1.5*x1 - 0.8*x2 + rnorm(n)
weights <- runif(n, 0.5, 2)
data <- data.frame(y = y, x1 = x1, x2 = x2)

# Basic usage with default priors
fit1 <- bqr.svy(y ~ x1 + x2, data = data, weights = weights)

# With informative priors
prior <- prior(
  p = 3,
  type = "univariate",
  beta_mean = c(2, 1.5, -0.8),
  beta_cov = diag(c(0.25, 0.25, 0.25)),
  sigma_shape = 3, sigma_rate = 2
)
fit2 <- bqr.svy(y ~ x1 + x2, data = data, weights = weights,
                method = "ald", prior = prior)

# Compare methods
fit_score <- bqr.svy(y ~ x1 + x2, data = data, weights = weights,
                     method = "score")
fit_approx <- bqr.svy(y ~ x1 + x2, data = data, weights = weights,
                      method = "approximate")
```

`mo.bqr.svy` *Multiple-Output Bayesian Quantile Regression for Complex Surveys (Directional EM)*

Description

Fits Bayesian quantile regression models for multivariate responses using the EM algorithm and a directional approach. The method projects the response into random unit vectors (directions) and their orthogonal complements, and then fits univariate Bayesian quantile regression models along each projection. The collection of fitted directions defines the multivariate quantile region.

Usage

```
mo.bqr.svy(
  formula,
  weights = NULL,
  data,
  quantile = 0.5,
  algorithm = "em",
  prior = NULL,
  n_dir = 1,
  epsilon = 1e-06,
```

```
  max_iter = 1000,
  verbose = FALSE,
  gamma_prior_var = 1e+06,
  ...
)
```

Arguments

formula	A formula object specifying the model.
weights	Optional vector of sampling weights. If NULL, equal weights are used.
data	A data frame containing the variables in the model.
quantile	Numeric vector of quantile levels (between 0 and 1, exclusive).
algorithm	Character string specifying the algorithm. Currently only "em" is supported.
prior	Prior specification. Can be: • NULL: Default priors are used for all quantiles • A single mo_bqr_prior object: Recycled for all quantiles • A list of mo_bqr_prior objects: One prior per quantile • A function f(tau, p, names): Generates quantile-specific priors
n_dir	Integer. Number of projection directions (if directions U are not supplied).
epsilon	Convergence tolerance for the EM algorithm.
max_iter	Maximum number of EM iterations.
verbose	Logical indicating whether to print progress messages.
gamma_prior_var	Numeric. Prior variance for the gamma coefficients associated with orthogonal complements.
...	Additional arguments for direction specification: **U** Optional user-specified matrix of directions ($d \times K$). If not provided, n_dir random unit vectors are generated automatically.

Details

The algorithm works by drawing or receiving as input a set of unit directions $u_k \in \mathbb{R}^d$. For each direction, an orthonormal basis of its orthogonal complement Γ_k is computed using pracma::nullspace. The response Y is then projected into the pair (u_k, Γ_k), and a Bayesian quantile regression is fitted along that direction using the EM algorithm. Results across all directions can be combined to approximate the multivariate quantile region.

Prior distributions can be specified globally or quantile-specific. When a list of priors is provided, elements can be named using either "q0.1" format or "0.1" format to match specific quantiles. When a function is provided, it will be called with (tau, p, names) for each quantile level.

Value

An object of class "mo.bqr.svy" containing:

call	The matched call
formula	The model formula
terms	The terms object
quantile	Vector of fitted quantiles

algorithm	Algorithm used
prior	List of priors used for each quantile
fit	List of fitted results for each quantile, each containing one sub-list per direction
coefficients	Coefficients from the first quantile
n_dir	Number of directions
U	Matrix of projection directions ($d \times K$)
Gamma_list	List of orthogonal complement bases, one per direction
n_obs	Number of observations
n_vars	Number of covariates
response_dim	Dimension of the response d

Examples

```
# Datos simulados para el ejemplo
set.seed(1)
n  <- 150
x1 <- runif(n,-1,1)
x2 <- rnorm(n)
y  <- 1 + 2*x1 + 0.5*x2 + rnorm(n)
mydata <- data.frame(y, x1, x2)

# Basic usage with default priors
fit1 <- mo.bqr.svy(y ~ x1 + x2, data = mydata,
                   quantile = c(0.1, 0.5, 0.9))

# Using quantile-specific priors via function
prior_fn <- function(tau, p, names) {
  variance <- ifelse(tau < 0.2 | tau > 0.8, 0.1, 1.0)
  mo_prior_default(p = p, beta_cov = diag(variance, p), names = names)
}
fit2 <- mo.bqr.svy(y ~ x1 + x2, data = mydata,
                   quantile = c(0.1, 0.5, 0.9), prior = prior_fn)

# Explicit control of directions
set.seed(1)
y1 <- 1 + 2*x1 + 0.5*x2 + rnorm(n)
y2 <- -1 + 1.5*x1 + rnorm(n)
y3 <- 0.5 - x2 + rnorm(n)
mydata <- data.frame(y1, y2, y3, x1, x2)
U <- matrix(rnorm(9), nrow = 3)  # d=3, K=2
U <- apply(U, 2, function(v) v / sqrt(sum(v^2))) # normalize
fit3 <- mo.bqr.svy(cbind(y1,y2,y3) ~ x1 + x2, data = mydata,
                   quantile = 0.5, U = U)
```

---

plot.bqr.svy *Plot Method for Bayesian Weighted Quantile Regression (Survey)*

Description

Plot method for objects of class `bqr.svy` produced by `bqr.svy()`. It can display fitted quantile curves, coefficient–quantile profiles, MCMC trace plots, and posterior densities.

Usage

```
## S3 method for class 'bqr.svy'
plot(
  x,
  y = NULL,
  type = c("fit", "quantile", "trace", "density"),
  predictor = NULL,
  tau = NULL,
  which = NULL,
  add_points = TRUE,
  combine = TRUE,
  show_ci = FALSE,
  ci_probs = c(0.1, 0.9),
  at = NULL,
  grid_length = 200,
  points_alpha = 0.4,
  point_size = 1.5,
  line_size = 1.2,
  main = NULL,
  use_ggplot = TRUE,
  theme_style = c("minimal", "classic", "bw", "light"),
  color_palette = c("viridis", "plasma", "set2", "dark2"),
  add_h0 = TRUE,
  add_ols = FALSE,
  ols_fit = NULL,
  ols_weights = NULL,
  ...
)

## S3 method for class 'bwqr_fit'
plot(x, ...)

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

Arguments

x	Object of class bqr.svy.
y	Ignored (S3 signature).
type	One of "fit", "quantile", "trace", "density".
predictor	(fit) Name of a numeric predictor; if NULL, the first numeric predictor (excluding the response) is used.
tau	Quantile(s) to plot; must appear in x$quantile. If NULL, all available are used.
which	(quantile/trace/density) Coefficient name or index to display.
add_points	(fit) Logical; overlay observed data points.
combine	(fit) Logical; if multiple tau: TRUE overlays curves in one panel; FALSE uses one panel per quantile.
show_ci	(fit) Logical; draw credible bands.
ci_probs	(fit) Length-2 numeric vector with lower/upper probabilities for credible bands.

at	(fit) Named list of fixed values for non-predictor covariates (see Details).
grid_length	(fit) Integer; number of points in the predictor grid.
points_alpha	(fit) Point transparency in [0,1].
point_size	(fit) Point size.
line_size	(fit/quantile) Line width for fitted/summary lines.
main	Optional main title.
use_ggplot	Logical; if TRUE, return a ggplot object.
theme_style	(ggplot) One of "minimal", "classic", "bw", "light".
color_palette	(ggplot) One of "viridis", "plasma", "set2", "dark2".
add_h0	(quantile) Logical; add a horizontal reference at $y = 0$.
add_ols	(quantile) Logical; add the OLS estimate (dotted line) for the selected coefficient.
ols_fit	(quantile) Optional precomputed lm object; if NULL, an lm() is fitted internally using x$model and x$terms.
ols_weights	(quantile) Optional numeric vector of weights when fitting OLS internally (length must match nrow(x$model)).
...	Accepted for compatibility; ignored by internal plotting code.

Details

Supported plot types:

- type = "fit": Fitted quantile curves versus a single numeric predictor. Optionally overlay observed points and credible bands. Other covariates can be held fixed via at.
- type = "quantile": A single coefficient as a function of the quantile τ. Optionally add a reference line at 0 and the corresponding OLS estimate.
- type = "trace": MCMC trace for one selected coefficient at a chosen τ.
- type = "density": Posterior density for one selected coefficient at a chosen τ.

Notes:

- tau must be included in x$quantile. If NULL, all available quantiles in the object are used.
- For type = "fit", predictor must be a numeric column in the original model. If NULL, the first numeric predictor (different from the response) is chosen automatically.
- For type = "fit", at is a named list (list(var = value, ...)) used to fix other covariates while plotting versus predictor. Provide valid levels for factors.
- When use_ggplot = TRUE, a ggplot object is returned and the appearance is controlled by theme_style and color_palette. Otherwise, base graphics are used and the function returns invisible(NULL).

Value

invisible(NULL) for base R graphics, or a ggplot object if use_ggplot = TRUE.

Examples

```
data(mtcars)
fit <- bqr.svy(mpg ~ wt + hp + cyl, data = mtcars,
               quantile = c(0.25, 0.5, 0.75), method = "ald",
               niter = 20000, burnin = 10000, thin = 5)

plot(fit, type = "fit", predictor = "wt", show_ci = TRUE)
plot(fit, type = "quantile", which = "wt", add_h0 = TRUE, add_ols = TRUE)
plot(fit, type = "trace", which = "wt", tau = 0.5)
plot(fit, type = "density", which = "wt", tau = 0.5)
```

---

print.tauBayesW *Print methods for tauBayesW objects*

Description

This page groups all S3 `print()` methods for the summary objects returned by **tauBayesW**: `summary.bqr.svy`, `summary.bwqr_fit`, and `summary.mo.bqr.svy`.

Usage

```
## S3 method for class 'summary.bqr.svy'
print(x, ...)

## S3 method for class 'summary.bwqr_fit'
print(x, ...)

## S3 method for class 'summary.mo.bqr.svy'
print(x, ...)

## S3 method for class 'summary.tauBayesW_multi'
print(x, ...)
```

---

prior *Unified Prior Specification for Bayesian Quantile Regression*

Description

A unified interface for creating prior distributions for both univariate (`bqr.svy`) and multivariate (`mo.bqr.svy`) Bayesian quantile regression models. This function automatically detects the model type and creates the appropriate prior object.

Usage

```
prior(
  p,
  type = NULL,
  beta_mean = rep(0, p),
  beta_cov = diag(1e+06, p),
  sigma_shape = 0.001,
  sigma_rate = 0.001,
  names = NULL
)
```

Arguments

p	Number of regression coefficients (including the intercept).
type	Character string specifying the model type: "MCMC" for bqr.svy models or "EM" for mo.bqr.svy models. If NULL (default), defaults to "MCMC".
beta_mean	Numeric vector of prior means for regression coefficients (length p). If a scalar is supplied, it is expanded to length p. Default is a vector of zeros.
beta_cov	Prior covariance matrix for regression coefficients. May be: • A p x p matrix • A scalar (expanded to diag(scalar, p)) • A length-p vector (expanded to diag(vector)) Default is diag(1e6, p) (vague prior).
sigma_shape	Shape parameter for the Inverse-Gamma prior on σ^2. Only used for ALD method in univariate models. Default is 0.001.
sigma_rate	Rate parameter for the Inverse-Gamma prior on σ^2. Only used for ALD method in univariate models. Default is 0.001.
names	Optional character vector of coefficient names to attach to the prior.

Details

This function provides a unified interface that replaces the need to know the specific prior creation functions for each model type.

For univariate models (type = "MCMC"):

- Uses parameters beta_mean, beta_cov, sigma_shape, sigma_rate
- Creates a bqr_prior object compatible with bqr.svy
- Sigma parameters are only used with method = "ald"

For multivariate models (type = "EM"):

- Uses parameters beta_mean, beta_cov, sigma_shape, sigma_rate
- Creates a mo_bqr_prior object compatible with mo.bqr.svy
- All parameters are used in the multivariate setting

Value

For univariate models: a bqr_prior object. For multivariate models: a mo_bqr_prior object.

See Also

bqr.svy, mo.bqr.svy, summary

Examples

```
# Univariate model priors (default)
prior_univ <- prior(p = 3)
prior_univ_info <- prior(
  p = 3,
  beta_mean = c(2, 1.5, -0.8),
  beta_cov = diag(c(0.25, 0.25, 0.25)),
  sigma_shape = 3,
  sigma_rate = 2
)

# Multivariate model priors
prior_mult <- prior(p = 3, type = "multivariate")
prior_mult_info <- prior(
  p = 3,
  type = "multivariate",
  beta_mean = c(0, 1, -0.5),
  beta_cov = diag(c(1, 1, 1))
)

# Usage in models
## Not run:
# Univariate
fit1 <- bqr.svy(y ~ x1 + x2, data = mydata, prior = prior_univ)

# Multivariate
fit2 <- mo.bqr.svy(cbind(y1, y2) ~ x1 + x2, data = mydata, prior = prior_mult)

## End(Not run)
```

---

summary.tauBayesW *Summary methods for tauBayesW objects*

Description

This page groups all S3 summary() methods provided by the **tauBayesW** package for classes bqr.svy, bwqr_fit, and mo.bqr.svy. Keeping them under one help page makes the manual concise while regular S3 dispatch still works as usual.

Posterior summary (means, credible intervals, R-hat, bulk/tail ESS) for objects of class bqr.svy. Supports one or multiple quantiles.

Posterior summary for bwqr_fit (single quantile). Computes means, SDs, R-hat, bulk/tail ESS, and credible intervals.

MAP-only summary per quantile and direction for mo.bqr.svy (no SD/CI).

If object is a *list* whose elements are tauBayesW fits (bwqr_fit, bqr.svy, mo.bqr.svy), this method computes each element's summary and returns a unified table. Otherwise it falls back to the next method.

Usage

```
## S3 method for class 'bqr.svy'
summary(object, probs = c(0.025, 0.975), digits = 3, ...)

## S3 method for class 'bwqr_fit'
summary(object, probs = c(0.025, 0.975), digits = 3, max_lag = 200, ...)

## S3 method for class 'mo.bqr.svy'
summary(object, digits = 4, ...)

## S3 method for class 'list'
summary(object, ..., methods = NULL, target_tau = 0.5, digits = 3)
```

Arguments

`object`	An object of class `mo.bqr.svy`.
`probs`	Credible interval probabilities.
`digits`	Number of decimals for printing the combined table.
`...`	Unused.
`max_lag`	Ignored (kept for backward compatibility).
`methods`	Optional character vector with labels for each fit.
`target_tau`	For `bqr.svy`, choose the block with τ closest to this value (default `0.5`).

Value

A `summary.bqr.svy` object containing one block per τ.

A `summary.bwqr_fit` object.

A `summary.mo.bqr.svy` object.

Index