Package 'SelectBoost.quantile'

April 13, 2026

Title 'SelectBoost'-Style Variable Selection for Quantile Regression

Date 2026-04-07

Version 0.3.1

Author Frederic Bertrand [cre, aut] (ORCID:
<https://orcid.org/0000-0002-0837-8281>)

Maintainer Frederic Bertrand <frederic.bertrand@lecnam.net>

Description A 'SelectBoost'-inspired workflow for sparse quantile regression. The package builds correlation neighborhoods, perturbs correlated predictors with a directional sampler inspired by the original 'SelectBoost' internals, refits penalized quantile regression models on the perturbed designs, and aggregates variable-selection frequencies across a path of correlation thresholds.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Imports graphics, movMF, quantreg, stats, utils, withr

Suggests knitr, pkgload, rmarkdown, testthat (>= 3.0.0)

URL https://fbertran.github.io/SelectBoost.quantile/,
https://github.com/fbertran/SelectBoost.quantile

BugReports https://github.com/fbertran/SelectBoost.quantile/issues

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2026-04-13 11:40:08 UTC

Contents

`benchmark_quantile_selection` *Benchmark quantile-selection methods on correlated designs*

Description

`benchmark_quantile_selection()` runs a reproducible simulation study over a set of scenarios and compares three selectors:

Usage

```
benchmark_quantile_selection(
  scenarios = default_quantile_benchmark_scenarios(),
  methods = c("lasso", "lasso_tuned", "selectboost"),
  replications = 20,
  threshold = 0.55,
  selection_metric = c("hybrid", "frequency"),
 selectboost_args = list(B = 20, step_num = 0.25, screen = "auto", tune_lambda = "cv",
   lambda_rule = "one_se", lambda_inflation = 1.25, nlambda = 12, folds = 5, repeats =
   1, subsamples = 25, sample_fraction = 0.5, complementary_pairs = TRUE, max_group_size
     = 15, verbose = FALSE),
 tuned_args = list(method = "cv", rule = "one_se", lambda_inflation = 1.25, nlambda =
    12, folds = 5, repeats = 1, verbose = FALSE),
  lasso_args = list(),
  standardize = TRUE,
  eps = 1e-06,
  seed = NULL,
  verbose = interactive()
)
```

Arguments

scenarios	Named list of scenario specifications. Each entry is passed to simulate_quantile_data(). Use default_quantile_benchmark_scenarios() for a ready-made validation grid.
methods	Methods to benchmark. Supported values are "lasso", "lasso_tuned", and "selectboost".
replications	Number of Monte Carlo replications per scenario.
threshold	Selection-frequency threshold used when extracting the stable support from selectboost_quantile().
selection_metric	Summary score used when extracting the stable support from selectboost_quantile(). "hybrid" combines stability and fitted effect size, while "frequency" uses mean path frequency alone.
selectboost_args	Additional named arguments passed to selectboost_quantile() for the "selectboost" method.
tuned_args	Additional named arguments passed to tune_lambda_quantile() for the "lasso_tuned" method.
lasso_args	Additional named arguments passed to quantile_lasso_selector() for the "lasso" method.
standardize	Should the lasso baselines use the same standardized design as selectboost_quantile()?
eps	Numerical tolerance used to turn coefficients into selections.
seed	Optional random seed.
verbose	Should progress messages be emitted?

Details

- "lasso": plain quantreg::rq.fit.lasso() support,
- "lasso_tuned": quantile lasso with tune_lambda_quantile(),
- "selectboost": stable support extracted from selectboost_quantile().

Each row in the returned benchmark table records support recovery, false discoveries, runtime, and failure status for one scenario, replication, and method.

Value

An object of class "benchmark_quantile_selection" with raw per-replication results in results.

Examples

```
scenarios <- default_quantile_benchmark_scenarios(
  tau = 0.5,
  regimes = "moderate_corr"
)
bench <- benchmark_quantile_selection(
  scenarios = scenarios,
  replications = 1,
```

```
  selectboost_args = list(B = 2, step_num = 1, tune_lambda = "bic", nlambda = 3),
  tuned_args = list(method = "bic", nlambda = 3),
  verbose = FALSE,
  seed = 1
)
summary(bench)
```

`coef.selectboost_quantile` *Extract coefficients from a SelectBoost-style quantile fit*

Description

Extract coefficients from a SelectBoost-style quantile fit

Usage

```
## S3 method for class 'selectboost_quantile'
coef(
  object,
  tau = NULL,
  c0 = min(object$c0_seq),
  threshold = NULL,
  include_intercept = TRUE,
  standardized = FALSE,
  ...
)
```

Arguments

`object`	A `selectboost_quantile()` fit.
`tau`	Optional quantile level to extract for multi-`tau` fits. When omitted, a named list is returned.
`c0`	Threshold along the perturbation path. The closest available `c0` value is used when `threshold` is not `NULL`.
`threshold`	Optional minimum selection frequency required for inclusion. When `NULL`, all baseline coefficients are returned.
`include_intercept`	Should the intercept be included?
`standardized`	Should coefficients be returned on the standardized model scale instead of the original predictor scale?
`...`	Unused.

Value

A named numeric vector or a named list of such vectors.

default_quantile_benchmark_scenarios *Default validation scenarios for quantile-selection benchmarks*

Description

`default_quantile_benchmark_scenarios()` returns a named list of simulation scenarios covering moderate and strong correlation, block dependence, high-dimensional designs, and misspecified noise. The output is designed to feed directly into `benchmark_quantile_selection()`.

Usage

```
default_quantile_benchmark_scenarios(
  tau = c(0.25, 0.5, 0.75),
 regimes = c("moderate_corr", "high_corr", "block_corr", "high_dim", "heavy_tail",
    "heteroskedastic")
)
```

Arguments

`tau`	Quantile levels to include in the validation grid. Each regime is expanded over these values.
`regimes`	Character vector selecting which regimes to include.

Value

A named list of scenario specifications.

Examples

```
scenarios <- default_quantile_benchmark_scenarios(
  tau = c(0.25, 0.5),
  regimes = c("moderate_corr", "heavy_tail")
)
names(scenarios)
```

group_neighbors *Grouping functions for SelectBoost.quantile*

Description

`group_neighbors()` reproduces the variable-wise neighborhood construction used by the original `SelectBoost::group_func_1()`: each variable is paired with the predictors whose absolute correlation exceeds `c0`.

Usage

```
group_neighbors(abs_corr, c0)

group_components(abs_corr, c0)
```

Arguments

abs_corr	Absolute correlation matrix.
c0	Correlation threshold in [0, 1].

Details

group_components() maps each variable to the connected component induced by the thresholded absolute correlation graph. This is a coarser grouping rule that can be useful for stress-testing the perturbation stage.

Value

A list of integer vectors, one neighborhood per variable.

---

plot.selectboost_quantile

Plot selection-frequency paths

Description

Plot selection-frequency paths

Usage

```
## S3 method for class 'selectboost_quantile'
plot(x, tau = NULL, vars = NULL, ...)
```

Arguments

x	A selectboost_quantile() fit.
tau	Optional quantile level to plot for multi-tau fits. Defaults to the first available tau.
vars	Optional subset of variables to plot. Defaults to the six variables with the highest mean selection frequency.
...	Passed to graphics::matplot().

Value

Invisibly returns the plotted frequency matrix.

predict.selectboost_quantile *Predict from a SelectBoost-style quantile fit*

Description

Predict from a SelectBoost-style quantile fit

Usage

```
## S3 method for class 'selectboost_quantile'
predict(
  object,
  newdata,
  tau = NULL,
  c0 = min(object$c0_seq),
  threshold = NULL,
  ...
)
```

Arguments

object	A selectboost_quantile() fit.
newdata	New data used for prediction. Required.
tau	Optional quantile level to predict for multi-tau fits. When omitted, predictions for all fitted tau values are returned.
c0	Threshold along the perturbation path. The closest available c0 value is used when threshold is not NULL.
threshold	Optional selection-frequency threshold used to zero-out unstable coefficients before prediction. When NULL, the full baseline fit is used.
...	Unused.

Value

A numeric vector for single-tau predictions or a matrix with one column per tau.

quantile_lasso_selector *Sparse quantile-regression selector*

Description

A thin wrapper around `quantreg::rq.fit.lasso()` that always includes an unpenalized intercept and returns a named coefficient vector.

Usage

```
quantile_lasso_selector(x, y, tau = 0.5, lambda = NULL, ...)
```

Arguments

`x`	Numeric design matrix.
`y`	Numeric response vector.
`tau`	Quantile level in `(0, 1)`.
`lambda`	Optional lasso penalty. A scalar applies the same penalty to every slope, while a vector may be supplied either for the slopes alone or for the full coefficient vector including the intercept.
`...`	Reserved for future selector variants.

Value

A named coefficient vector.

selectboost_quantile *SelectBoost-style quantile regression*

Description

`selectboost_quantile()` adapts the core SelectBoost workflow to sparse quantile regression:

Usage

```
selectboost_quantile(
  x,
  y = NULL,
  tau = 0.5,
  B = 50,
  c0_seq = NULL,
  step_num = 0.1,
  group = group_neighbors,
```

```
  max_group_size = NULL,
  screen = c("auto", "none", "quantile_rank"),
  screen_size = NULL,
  lambda = NULL,
  tune_lambda = c("none", "cv", "bic"),
  lambda_rule = c("min", "one_se"),
  lambda_factors = NULL,
  lambda_inflation = 1,
  nlambda = 20,
  lambda_min_ratio = 0.05,
  folds = 5,
  repeats = 1,
  subsamples = 1,
  sample_fraction = 0.5,
  complementary_pairs = FALSE,
  selector = quantile_lasso_selector,
  standardize = TRUE,
  eps = 1e-06,
  seed = NULL,
  data = NULL,
  subset = NULL,
  na.action = stats::na.fail,
  verbose = interactive(),
  ...
)
```

Arguments

x	Numeric design matrix or a formula.
y	Numeric response vector when x is a matrix.
tau	Quantile level in (0, 1). Can be a vector.
B	Number of perturbation replicates for each c0 threshold.
c0_seq	Optional decreasing sequence of correlation thresholds. When NULL, it is computed from empirical correlation quantiles using step_num.
step_num	Step size used to build the default c0 path.
group	Grouping rule used to convert the absolute correlation matrix and threshold c0 into a list of neighborhoods, one per variable. Can be a function or the name of one. Functions must accept (abs_corr, c0).
max_group_size	Optional cap on the size of each correlation neighborhood. When supplied, only the strongest absolute correlations are retained within each variable's group.
screen	Screening rule applied before the SelectBoost loop. "auto" enables tau-aware rank screening when p > n, "none" disables screening, and "quantile_rank" always uses the built-in rank-score screen. Functions must accept (x, y, tau, screen_size).
screen_size	Optional number of predictors retained after screening.
lambda	Optional lasso penalty supplied to quantreg::rq.fit.lasso(). A scalar applies a common slope penalty, while a full penalty vector can also be supplied.

When tau has length greater than one, lambda can also be a list with one entry per tau.

tune_lambda — One of "none", "cv", or "bic". When not "none", the package tunes a penalty profile once on the original design and reuses it for all perturbations.

lambda_rule — Selection rule used after tuning. "min" takes the best tuning score, while "one_se" applies the one-standard-error rule when tune_lambda = "cv".

lambda_factors — Optional positive multipliers applied to the default quantile-lasso penalty profile during tuning.

lambda_inflation — Optional multiplier applied after tuning to favor a stronger selection penalty.

nlambda — Number of tuning candidates when lambda_factors is NULL.

lambda_min_ratio — Smallest tuning multiplier used to generate the default tuning grid.

folds — Number of cross-validation folds when tune_lambda = "cv".

repeats — Number of repeated fold assignments when tune_lambda = "cv".

subsamples — Number of subsample draws used for stability selection. Values greater than one aggregate selection frequencies across subsamples.

sample_fraction — Fraction of observations drawn in each subsample when subsamples > 1.

complementary_pairs — Should subsamples be generated as complementary pairs?

selector — Function used to fit the sparse quantile model. It must accept (x, y, tau, lambda, ...) and return a named coefficient vector including an intercept.

standardize — Should the selector be fitted on the SelectBoost-normalized design? When TRUE, columns are centered and scaled to unit Euclidean norm before fitting, matching the original package. When FALSE, perturbations are still generated in the normalized space but mapped back to the original scale before model fitting.

eps — Numerical tolerance used to turn coefficients into selections.

seed — Optional random seed for reproducible perturbations and tuning.

data — Optional data frame used when x is a formula.

subset — Optional subset expression used with the formula interface.

na.action — Missing-data handler used with the formula interface.

verbose — Should the routine report progress?

... — Additional arguments forwarded to selector.

Details

1. build a centered, unit-norm design as in SelectBoost::boost.normalize(),
2. compute correlation neighborhoods along a c0 path,
3. fit a directional distribution to each variable's sign-aligned neighborhood in the sample hyperplane,
4. draw perturbed predictors from those fitted directional models,
5. refit penalized quantile regression and aggregate selection frequencies.

This version keeps the public API stable while separating the internals into explicit preprocessing, grouping, directional perturbation, and tuning stages.

Value

An object of class "selectboost_quantile" with components: frequencies, baseline, baseline_standardized, c0_seq, tau, B, lambda, lambda_tuning, call, and preprocessing metadata.

Examples

```
sim <- simulate_quantile_data(n = 80, p = 12, active = 1:3, seed = 1)
fit <- selectboost_quantile(sim$x, sim$y, tau = 0.5, B = 8, seed = 1)
print(fit)
summary(fit, threshold = 0.6)

dat <- data.frame(y = sim$y, sim$x)
fit_formula <- selectboost_quantile(
  y ~ .,
  data = dat,
  tau = 0.5,
  B = 4,
  step_num = 0.5,
  seed = 1
)
```

---

simulate_quantile_data

Simulate a sparse quantile-regression problem

Description

Simulate a sparse quantile-regression problem

Usage

```
simulate_quantile_data(
  n = 200,
  p = 40,
  active = 1:5,
  beta = c(2, 1.5, -1.5, 1, -1),
  tau = 0.5,
  rho = 0.7,
  correlation = c("toeplitz", "block"),
  block_size = 5L,
  error = c("gaussian", "student", "laplace", "heteroskedastic"),
  error_df = 3,
  heteroskedastic_strength = 0.75,
  seed = NULL
)
```

Arguments

n	Number of observations.
p	Number of predictors.
active	Indices of active predictors.
beta	Coefficients for the active predictors. Recycled as needed.
tau	Quantile level whose conditional linear predictor is controlled.
rho	Toeplitz correlation parameter for the predictors.
correlation	Correlation structure. One of "toeplitz" or "block".
block_size	Block size used when correlation = "block".
error	Error distribution. One of "gaussian", "student", "laplace", or "heteroskedastic".
error_df	Degrees of freedom when error = "student".
heteroskedastic_strength	Positive scale multiplier used when error = "heteroskedastic".
seed	Optional random seed.

Value

A list containing x, y, beta, active, tau, and the simulation settings used to generate the data.

Examples

```
sim <- simulate_quantile_data(seed = 42)
str(sim, max.level = 1)
```

---

summary.benchmark_quantile_selection

Summarize a quantile-selection benchmark

Description

Summarize a quantile-selection benchmark

Usage

```
## S3 method for class 'benchmark_quantile_selection'
summary(object, ...)
```

Arguments

object	A benchmark_quantile_selection() object.
...	Unused.

Value

A data frame with one row per scenario, quantile level, and method.

summary.selectboost_quantile *Summarize a SelectBoost-style quantile fit*

Description

Summarize a SelectBoost-style quantile fit

Usage

```
## S3 method for class 'selectboost_quantile'
summary(
  object,
  threshold = 0.55,
  tau = NULL,
  enforce_monotone = TRUE,
  selection_metric = c("hybrid", "frequency"),
  ...
)
```

Arguments

object	A selectboost_quantile() fit.
threshold	Frequency threshold used to define the reported stable support.
tau	Optional quantile level to summarize when the fit contains multiple tau values. When omitted, a multi-summary is returned.
enforce_monotone	Should the frequency paths be post-processed into a non-increasing function of the perturbation strength?
selection_metric	Summary score used to define the stable support. "frequency" thresholds the pathwise mean selection frequency, while "hybrid" downweights frequently selected variables whose fitted baseline effect size remains weak.
...	Unused.

Value

An object of class "summary.selectboost_quantile" or "summary.selectboost_quantile_multi".

`support_selectboost_quantile` *Extract selected support at a frequency threshold*

Description

Extract selected support at a frequency threshold

Usage

```
support_selectboost_quantile(
  object,
  tau = NULL,
  c0 = min(object$c0_seq),
  threshold = 0.55,
  selection_metric = c("hybrid", "frequency"),
  include_intercept = FALSE
)
```

Arguments

`object`	A `selectboost_quantile()` fit.
`tau`	Optional quantile level to extract for multi-`tau` fits. When omitted, a named list is returned.
`c0`	Threshold along the perturbation path. The closest available `c0` value is used when `selection_metric = "frequency"`.
`threshold`	Minimum summary score required for inclusion.
`selection_metric`	Support score used to define the returned support. `"hybrid"` reuses the summary-level hybrid stability/effect-size score, while `"frequency"` applies the threshold directly to the selection frequency at the requested `c0`.
`include_intercept`	Should the intercept be included in the returned support?

Value

A character vector or a named list of character vectors.

`tune_lambda_quantile` *Tune the lasso penalty for sparse quantile regression*

Description

`tune_lambda_quantile()` tunes a penalty profile once on the original design and returns the selected penalty vector. The default grid rescales the `quantreg::LassoLambdaHat()` profile rather than using a single scalar, which keeps the tuning step aligned with the underlying quantile-lasso routine.

Usage

```
tune_lambda_quantile(
  x,
  y = NULL,
  tau = 0.5,
  method = c("cv", "bic"),
  rule = c("min", "one_se"),
  lambda_factors = NULL,
  lambda_inflation = 1,
  nlambda = 20,
  lambda_min_ratio = 0.05,
  folds = 5,
  repeats = 1,
  selector = quantile_lasso_selector,
  standardize = TRUE,
  eps = 1e-06,
  seed = NULL,
  data = NULL,
  subset = NULL,
  na.action = stats::na.fail,
  verbose = interactive(),
  ...
)
```

Arguments

`x`	Numeric design matrix or a formula.
`y`	Numeric response vector when `x` is a matrix.
`tau`	Quantile level in `(0, 1)`. Can be a vector.
`method`	One of `"cv"` or `"bic"`.
`rule`	Selection rule for choosing the tuned penalty from the candidate grid. `"min"` takes the minimizer of the tuning score, while `"one_se"` applies the one-standard-error rule when `method = "cv"`.
`lambda_factors`	Optional positive multipliers applied to the default penalty profile.

`lambda_inflation`
: Optional multiplier applied after tuning to enforce a stronger penalty for selection than for prediction.

`nlambda`
: Number of tuning candidates when `lambda_factors` is `NULL`.

`lambda_min_ratio`
: Smallest multiplier in the default grid.

`folds`
: Number of folds when `method = "cv"`.

`repeats`
: Number of repeated fold assignments when `method = "cv"`.

`selector`
: Function used to fit the sparse quantile model.

`standardize`
: Should tuning use the SelectBoost-normalized design?

`eps`
: Numerical tolerance used to count active coefficients for the BIC heuristic.

`seed`
: Optional random seed.

`data`
: Optional data frame used when `x` is a formula.

`subset`
: Optional subset expression used with the formula interface.

`na.action`
: Missing-data handler used with the formula interface.

`verbose`
: Should tuning report progress?

`...`
: Additional arguments forwarded to `selector`.

Value

An object of class `"tuned_lambda_quantile"`.

Examples

```
sim <- simulate_quantile_data(n = 60, p = 10, active = 1:3, seed = 2)
tuned <- tune_lambda_quantile(
  sim$x,
  sim$y,
  tau = 0.5,
  method = "bic",
  nlambda = 6
)
tuned$factor
```

Index