Package: rbridge (via r-universe)

October 13, 2026

Type Package

Title Restricted Bridge Estimation

Version 1.0.4

Date 2026-10-12

Description Implements bridge regression with linear coefficient restrictions using the local quadratic approximation described by Yuzbasi, Arashi and Akdeniz (2021) <doi:10.1007/s00500-021-05763-9>. Provides coefficient paths, cross-validation, prediction and coefficient extraction for restricted and unrestricted linear models.

License GPL-3

Imports Rcpp, Matrix, methods, graphics, stats, utils

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.3

NeedsCompilation yes

URL https://byuzbasi.github.io/rbridge/, https://github.com/byuzbasi/rbridge

BugReports https://github.com/byuzbasi/rbridge/issues

Author Bahadir Yuzbasi [aut, cre], Mohammad Arashi [aut], Fikri Akdeniz [aut], Jerome Friedman [ctb] (Utility functions adapted from glmnet), Trevor Hastie [ctb] (Utility functions adapted from glmnet), Robert Tibshirani [ctb] (Utility functions adapted from glmnet)

Maintainer Bahadir Yuzbasi <b.yzb@hotmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-13 08:30:08 UTC

RemoteUrl https://github.com/cran/rbridge

RemoteRef HEAD

RemoteSha ec1da56c37f83778515eb3e787edb03229bfb83c

Contents

bridge 2
coef.bridge 5
coef.cv.bridge 6
coef.cv.rbridge 7
coef.rbridge 9
cv.bridge 10
cv.rbridge 13
plot.cv.bridge 16
plot.cv.rbridge 18
predict.bridge 19
predict.cv.bridge 21
predict.cv.rbridge 22
predict.rbridge 24
rbridge 26
rbridge_info 29

Index **31**

`bridge` *Fit a bridge coefficient path*

Description

Fit a bridge-penalized least-squares path using local quadratic approximation.

Usage

```
bridge(
  X,
  y,
  q = 1,
  lambda.min = ifelse(n > p, 0.001, 0.05),
  nlambda = 100,
  lambda,
  eta = 1e-07,
  converge = 10^10
)
```

Arguments

`X`	Numeric design matrix with `n` observation rows and `p` predictor columns. Supply finite values and predictors with nonzero variation; omit an intercept column. A data frame is converted using `model.matrix(~0+., X)`; restrictions and future predictions must use the resulting predictor order.
`y`	Numeric response vector of length `n`, containing finite values.
`q`	Bridge-penalty exponent, a finite number in `(0, 2]`. Values 1 and 2 specify the L1 and L2 penalty forms. Estimation uses local quadratic approximation (LQA).
`lambda.min`	Positive absolute endpoint of the automatically generated penalty grid. The default is `0.001` when `n > p`, and `0.05` otherwise. This argument is a penalty value, not a ratio. In CV results, the field also named `lambda.min` instead denotes the selected penalty.
`nlambda`	Number of automatically generated penalty values, an integer of at least 2. The default is `100`; ignored when `lambda` is supplied.
`lambda`	Optional nonempty numeric vector of finite, nonnegative penalty values. Its order is retained. When supplied, it determines the fitted path and overrides `nlambda` and the grid argument `lambda.min`.
`eta`	Positive numerical tolerance, with default `1e-7`. Used for coefficient elimination and stopping on squared coefficient changes.
`converge`	Nonnegative initial convergence-distance value, retained for compatibility. The default is `1e10`. Iteration stops when the updated squared coefficient change is at most `eta`, or at the historical cap.

Details

Predictors are centered and divided by their centered root-mean-square (RMS) deviations, using the observation count `n` as the divisor. The response is centered. The intercept is reconstructed after fitting, and returned coefficients are expressed on the original predictor scale. The generated penalties are logarithmically spaced between an internally computed starting value and the absolute endpoint `lambda.min`.

Let Z be the n by p centered, RMS-standardized design, y_c the centered response vector of length n, and b the p-dimensional standardized slope vector. For exponent $q \in (0, 2]$ and penalty weight $\lambda \geq 0$, Equation (1) defines the penalized least-squares target:

$$\min_b \left\{ \|y_c - Zb\|_2^2 + \lambda \sum_{j=1}^{p} |b_j|^q \right\}. \qquad (1)$$

In Equation (1), $\|\cdot\|_2^2$ is the squared Euclidean norm, so its first term is the unnormalized residual sum of squares. The index j runs over the predictors; the second term is the bridge penalty, with λ controlling its weight relative to residual error.

The fitting routine applies the article's LQA updates, coefficient elimination and stopping rules. The criterion is nonconvex when $q < 1$, so the iterative result is not a guarantee of a global minimum. The getting-started vignette explains the penalty geometry and the fixed-LQA restriction correction.

Value

A list of class `bridge` with components:

`betas`	Numeric coefficient matrix with `p + 1` rows and `length(lambda)` columns, one column per fitted penalty. The first row is `(Intercept)`; the remaining rows are original-scale slopes in predictor order. Column names identify the fitted penalties.
`lambda`	Numeric vector of penalties, in fitted column order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

References

Yuzbasi, B., Arashi, M. and Akdeniz, F. (2021). Penalized regression via the restricted bridge estimator. Soft Computing, 25, 8401–8416. doi:10.1007/s00500021057639.

See Also

`cv.bridge`, `coef.bridge`, `predict.bridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
# L1 and L2 penalty forms on the same observations.
model1 <- bridge(X, y, q = 1, nlambda = 5)
model2 <- bridge(X, y, q = 2, nlambda = 5)
dim(model1$betas)
# Extract one fitted column and predict five observations.
selected <- model1$lambda[2]
coef(model1, s = selected)
predict(model1, newx = X[1:5, , drop = FALSE], s = selected)
```

coef.bridge *Extract coefficients from a fitted bridge object*

Description

Extract original-scale intercept and slope coefficients from a fitted bridge object.

Usage

```
## S3 method for class 'bridge'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

`object`	Fitted object returned by `bridge`, with `p` predictors in its training design.
`s`	Numeric fitted penalty value(s). Values must match `object$lambda` exactly; no interpolation is used. Omitting this argument returns all fitted penalties. The character selector `lambda.min` requests the smallest fitted penalty. Use a CV object for the selector `lambda.1se`.
`...`	Additional arguments accepted for compatibility.

Details

The returned matrix retains its dimensions for a single selected penalty.

Value

Numeric matrix with `p + 1` rows and one column per selected penalty, even when only one penalty is selected. The first row is `(Intercept)`; subsequent rows are original-scale slopes in the training predictor order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`predict.bridge`, `bridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
```

```
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
model1 <- bridge(X, y, q = 1, nlambda = 5)
# A single selected penalty still gives a coefficient matrix.
coefficients <- coef(model1, s = model1$lambda[2])
coefficients
dim(coefficients)
# Omitting s extracts the full fitted coefficient path.
dim(coef(model1))
```

`coef.cv.bridge` *Extract coefficients from a fitted cv.bridge object*

Description

Extract original-scale intercept and slope coefficients from a fitted cross-validated cv.bridge object.

Usage

```
## S3 method for class 'cv.bridge'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

`object`	Fitted object returned by `cv.bridge`, with `p` predictors in its training design.
`s`	Penalty selection, using numeric fitted values or a character name. Numeric values must match `object$lambda` exactly; no interpolation is used. The character selectors are: `"lambda.min"` Select the penalty with minimum CV error. `"lambda.1se"` Select using the historical CV error bars. The default is `lambda.1se`.
`...`	Additional arguments accepted for compatibility.

Details

The returned matrix retains its dimensions for a single selected penalty. The defaults differ between `coef()` and `predict()`: coefficient extraction uses `lambda.1se`, whereas response prediction uses `lambda.min`. Specify the same `s` in both calls when comparing coefficients and their predictions.

Value

Numeric matrix with `p + 1` rows and one column per selected penalty, even when only one penalty is selected. The first row is `(Intercept)`; subsequent rows are original-scale slopes in the training predictor order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

predict.cv.bridge, cv.bridge

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.bridge(X, y, q = 1, nlambda = 5,
                    nfolds = 3, num_threads = 1)
# A single selected penalty still gives a coefficient matrix.
coefficients <- coef(model1, s = "lambda.min")
coefficients
dim(coefficients)
# The default uses the historical error-bar selector.
coef(model1)
```

`coef.cv.rbridge` *Extract coefficients from a fitted cv.rbridge object*

Description

Extract original-scale intercept and slope coefficients from a fitted cross-validated cv.rbridge object.

Usage

```
## S3 method for class 'cv.rbridge'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

`object`	Fitted object returned by cv.rbridge, with p predictors in its training design.
`s`	Penalty selection, using numeric fitted values or a character name. Numeric values must match `object$lambda` exactly; no interpolation is used. The character selectors are:
	`"lambda.min"` Select the penalty with minimum CV error.

	`"lambda.1se"` Select using the historical CV error bars.
	The default is `lambda.1se`.
`...`	Additional arguments accepted for compatibility.

Details

The returned matrix retains its dimensions for a single selected penalty. The defaults differ between `coef()` and `predict()`: coefficient extraction uses `lambda.1se`, whereas response prediction uses `lambda.min`. Specify the same `s` in both calls when comparing coefficients and their predictions. Returned slopes are on the original predictor scale; the linear restrictions concern standardized slopes. See `rbridge` for converting scales before checking restriction residuals.

Value

Numeric matrix with `p + 1` rows and one column per selected penalty, even when only one penalty is selected. The first row is `(Intercept)`; subsequent rows are original-scale slopes in the training predictor order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`predict.cv.rbridge`, `cv.rbridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5,
                     nfolds = 3, num_threads = 1)
# A single selected penalty still gives a coefficient matrix.
coefficients <- coef(model1, s = "lambda.min")
coefficients
dim(coefficients)
# The default uses the historical error-bar selector.
coef(model1)
```

`coef.rbridge` *Extract coefficients from a fitted rbridge object*

Description

Extract original-scale intercept and slope coefficients from a fitted rbridge object.

Usage

```
## S3 method for class 'rbridge'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

`object`	Fitted object returned by `rbridge`, with `p` predictors in its training design.
`s`	Numeric fitted penalty value(s). Values must match `object$lambda` exactly; no interpolation is used. Omitting this argument returns all fitted penalties. The character selector `lambda.min` requests the smallest fitted penalty. Use a CV object for the selector `lambda.1se`.
`...`	Additional arguments accepted for compatibility.

Details

The returned matrix retains its dimensions for a single selected penalty. Returned slopes are on the original predictor scale; the linear restrictions concern standardized slopes. See `rbridge` for converting scales before checking restriction residuals.

Value

Numeric matrix with `p + 1` rows and one column per selected penalty, even when only one penalty is selected. The first row is `(Intercept)`; subsequent rows are original-scale slopes in the training predictor order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`predict.rbridge`, `rbridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5)
# A single selected penalty still gives a coefficient matrix.
coefficients <- coef(model1, s = model1$lambda[2])
coefficients
dim(coefficients)
# Omitting s extracts the full fitted coefficient path.
dim(coef(model1))
```

---

cv.bridge *Cross-validation for bridge regression*

Description

Evaluate a fitted penalty path using contiguous validation folds and return the full-data coefficients together with two penalty selectors.

Usage

```
cv.bridge(
  X,
  y,
  q,
  lambda,
  nfolds = 10,
  lambda.min = ifelse(n > p, 0.001, 0.05),
  nlambda = 100,
  eta = 1e-07,
  converge = 10^10,
  num_threads = 10
)
```

Arguments

X	Numeric design matrix with n observation rows and p predictor columns. Supply finite values and predictors with nonzero variation; omit an intercept column. A data frame is converted using `model.matrix(~0+., X)`; restrictions and future predictions must use the resulting predictor order.
y	Numeric response vector of length n, containing finite values.
q	Bridge-penalty exponent, a finite number in `(0, 2]`. Values 1 and 2 specify the L1 and L2 penalty forms. Estimation uses local quadratic approximation (LQA).
lambda	Optional nonempty numeric vector of finite, nonnegative penalty values. Its order is retained. When supplied, it determines the fitted path and overrides nlambda and the grid argument `lambda.min`.
nfolds	Number of validation folds, an integer between 2 and n. The default is `10`.
lambda.min	Positive absolute endpoint of the automatically generated penalty grid. The default is `0.001` when n > p, and `0.05` otherwise. This argument is a penalty value, not a ratio. In CV results, the field also named `lambda.min` instead denotes the selected penalty.
nlambda	Number of automatically generated penalty values, an integer of at least 2. The default is `100`; ignored when `lambda` is supplied.
eta	Positive numerical tolerance, with default `1e-7`. Used for coefficient elimination and stopping on squared coefficient changes.
converge	Nonnegative initial convergence-distance value, retained for compatibility. The default is `1e10`. Iteration stops when the updated squared coefficient change is at most `eta`, or at the historical cap.
num_threads	Requested fold-level workers, a positive integer. The default is `10`; the effective count is capped at `nfolds`. Builds without OpenMP execute folds sequentially. Use `rbridge_info()` to inspect the compiled support.

Details

Fold assignment follows the supplied row order. The `nfolds` validation sets are contiguous blocks; their sizes can differ by one. Centering and RMS scaling use all observations before fold allocation, and each fold fit also reconstructs its training-set intercept. These historical preprocessing and error-bar conventions are retained. The reported validation error aggregates held-out predictions at each fitted penalty, and coefficients in `betas` come from a final fit to all observations.

For one fixed penalty, let K be `nfolds`, n the total observation count, I_k the validation indices in fold k, and e_i the held-out response residual for observation i. Equation (1) defines a fold's error contribution d_k, the pooled validation MSE, and the historical error-bar width:

$$
\begin{aligned}
d_k &= \frac{1}{n} \sum_{i \in I_k} e_i^2, \\
\mathrm{cve} &= \sum_{k=1}^{K} d_k, \\
\mathrm{cvse} &= \sqrt{\frac{1}{K(n-1)} \sum_{k=1}^{K} (d_k - \mathrm{cve})^2}.
\end{aligned} \tag{1}
$$

The folds partition all observations, so the sum in Equation (1) weights each held-out observation equally. The final quantity supplies the plotted bar width and the historical `lambda.1se` threshold. Its centering uses the pooled `cve`, not the average fold contribution, and it is not a conventional foldwise standard error.

`predict()` defaults to the minimum-error penalty `lambda.min`. `coef()` defaults to `lambda.1se`. Supply `s` explicitly to use the same penalty for both operations. Use `plot()` to inspect the curve, error bars and selected penalties.

Value

A list of class `cv.bridge` with components:

`cve`	Numeric vector of validation mean squared errors, one per penalty. Squared holdout residuals are summed across folds and divided by the total observation count, so each observation has the same weight.
`cvse`	Numeric vector from the historical CV error-bar calculation. It is not the conventional standard error of foldwise mean squared errors.
`cvup`, `cvlo`	Upper and lower curves, computed as `cve + cvse` and `cve - cvse`.
`lambda`	Numeric penalty vector in fitted order.
`nz`	Number of nonzero slopes, excluding the intercept, in each full-data fit.
`betas`	Full-data coefficient matrix with `p + 1` rows and `length(lambda)` columns. The first row is `(Intercept)`; the remaining rows are original-scale slopes in predictor order.
`lambda.min`	Fitted penalty with the smallest `cve`. Ties select the largest penalty among the minimizers. This selected value is distinct from the input argument specifying the generated grid endpoint.
`lambda.1se`	Largest fitted penalty whose `cve` is no greater than `cve + cvse` evaluated at the selected `lambda.min`. The name follows the historical convention; the threshold uses the package's `cvse` calculation described above.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`bridge`, `plot.cv.bridge`, `coef.cv.bridge`, `predict.cv.bridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
```

```
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
# Three contiguous folds and one worker keep this demonstration small.
model1 <- cv.bridge(X, y, q = 1,
                    nlambda = 5, nfolds = 3, num_threads = 1)
model2 <- cv.bridge(X, y, q = 2,
                    nlambda = 5, nfolds = 3, num_threads = 1)
# Inspect both selectors and explicitly choose the minimum-error fit.
c(minimum_error = model1$lambda.min, historical_1se = model1$lambda.1se)
coef(model1, s = "lambda.min")
predict(model1, newx = X[1:5, , drop = FALSE], s = "lambda.min")
plot(model1)
```

`cv.rbridge` *Cross-validation for restricted bridge regression*

Description

Evaluate a fitted penalty path using contiguous validation folds and return the full-data coefficients together with two penalty selectors.

Usage

```
cv.rbridge(
  X,
  y,
  q,
  R,
  r,
  lambda,
  nfolds = 10,
  lambda.min = ifelse(n > p, 0.001, 0.05),
  nlambda = 100,
  eta = 1e-07,
  converge = 10^10,
  num_threads = 10
)
```

Arguments

`X`	Numeric design matrix with `n` observation rows and `p` predictor columns. Supply finite values and predictors with nonzero variation; omit an intercept column. A data frame is converted using `model.matrix(~0+., X)`; restrictions and future predictions must use the resulting predictor order.
`y`	Numeric response vector of length `n`, containing finite values.
`q`	Bridge-penalty exponent, a finite number in `(0, 2]`. Values 1 and 2 specify the L1 and L2 penalty forms. Estimation uses local quadratic approximation (LQA).

R	Finite numeric matrix with m linearly independent restriction rows and p predictor columns. Each row specifies a linear combination of the standardized predictor coefficients, in the order of X. The intercept is excluded.
r	Finite numeric target vector of length m, or an m by 1 matrix. The targets have the units of the response when R is dimensionless. See Details for the coefficient scale and a residual check.
lambda	Optional nonempty numeric vector of finite, nonnegative penalty values. Its order is retained. When supplied, it determines the fitted path and overrides nlambda and the grid argument lambda.min.
nfolds	Number of validation folds, an integer between 2 and n. The default is 10.
lambda.min	Positive absolute endpoint of the automatically generated penalty grid. The default is 0.001 when n > p, and 0.05 otherwise. This argument is a penalty value, not a ratio. In CV results, the field also named lambda.min instead denotes the selected penalty.
nlambda	Number of automatically generated penalty values, an integer of at least 2. The default is 100; ignored when lambda is supplied.
eta	Positive numerical tolerance, with default 1e-7. Used for coefficient elimination and stopping on squared coefficient changes.
converge	Nonnegative initial convergence-distance value, retained for compatibility. The default is 1e10. Iteration stops when the updated squared coefficient change is at most eta, or at the historical cap.
num_threads	Requested fold-level workers, a positive integer. The default is 10; the effective count is capped at nfolds. Builds without OpenMP execute folds sequentially. Use rbridge_info() to inspect the compiled support.

Details

Fold assignment follows the supplied row order. The nfolds validation sets are contiguous blocks; their sizes can differ by one. Centering and RMS scaling use all observations before fold allocation, and each fold fit also reconstructs its training-set intercept. These historical preprocessing and error-bar conventions are retained. The reported validation error aggregates held-out predictions at each fitted penalty, and coefficients in betas come from a final fit to all observations.

For one fixed penalty, let K be nfolds, n the total observation count, I_k the validation indices in fold k, and e_i the held-out response residual for observation i. Equation (1) defines a fold's error contribution d_k, the pooled validation MSE, and the historical error-bar width:

$$
\begin{aligned}
d_k &= \frac{1}{n} \sum_{i \in I_k} e_i^2, \\
\mathrm{cve} &= \sum_{k=1}^{K} d_k, \\
\mathrm{cvse} &= \sqrt{\frac{1}{K(n-1)} \sum_{k=1}^{K} (d_k - \mathrm{cve})^2}. \qquad (1)
\end{aligned}
$$

The folds partition all observations, so the sum in Equation (1) weights each held-out observation equally. The final quantity supplies the plotted bar width and the historical lambda.1se threshold.

Its centering uses the pooled cve, not the average fold contribution, and it is not a conventional foldwise standard error.

predict() defaults to the minimum-error penalty lambda.min. coef() defaults to lambda.1se. Supply s explicitly to use the same penalty for both operations. Use plot() to inspect the curve, error bars and selected penalties.

Restrictions act on standardized slopes. To check a fitted original-scale slope vector, multiply it elementwise by the centered RMS deviations of the predictors before applying R. The resulting restriction residual is the fitted linear combination minus r; it should be examined with a numerical tolerance, particularly after coefficient thresholding. For example, a zero restriction on one slope has the same meaning on either scale, whereas a nonzero sum or contrast depends on the scales.

Value

A list of class cv.rbridge with components:

cve	Numeric vector of validation mean squared errors, one per penalty. Squared holdout residuals are summed across folds and divided by the total observation count, so each observation has the same weight.
cvse	Numeric vector from the historical CV error-bar calculation. It is not the conventional standard error of foldwise mean squared errors.
cvup, cvlo	Upper and lower curves, computed as cve + cvse and cve - cvse.
lambda	Numeric penalty vector in fitted order.
nz	Number of nonzero slopes, excluding the intercept, in each full-data fit.
betas	Full-data coefficient matrix with p + 1 rows and length(lambda) columns. The first row is (Intercept); the remaining rows are original-scale slopes in predictor order.
lambda.min	Fitted penalty with the smallest cve. Ties select the largest penalty among the minimizers. This selected value is distinct from the input argument specifying the generated grid endpoint.
lambda.1se	Largest fitted penalty whose cve is no greater than cve + cvse evaluated at the selected lambda.min. The name follows the historical convention; the threshold uses the package's cvse calculation described above.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

rbridge, plot.cv.rbridge, coef.cv.rbridge, predict.cv.rbridge

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
```

```
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)
# A second standardized-slope contrast: -b[1] + b[2] + b[5] = 0.5.
R2.mat <- matrix(c(-1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r2.vec <- matrix(0.5, nrow = 1, ncol = 1)
# Impose both independent restrictions together.
R3.mat <- rbind(R1.mat, R2.mat)
r3.vec <- matrix(c(6.5, 0.5), ncol = 1)
# Alternatively, set the five inactive generating slopes to zero.
p.active <- which(beta != 0)
R4.mat <- diag(p)[-p.active, , drop = FALSE]
r4.vec <- matrix(rep(0, p - length(p.active)), ncol = 1)

# Five penalties keep this help example fast; the default is 100.
# Three contiguous folds and one worker keep this demonstration small.
model1 <- cv.rbridge(X, y, q = 1, R = R1.mat, r = r1.vec,
                     nlambda = 5, nfolds = 3, num_threads = 1)
model2 <- cv.rbridge(X, y, q = 1, R = R2.mat, r = r2.vec,
                     nlambda = 5, nfolds = 3, num_threads = 1)
model3 <- cv.rbridge(X, y, q = 1, R = R3.mat, r = r3.vec,
                     nlambda = 5, nfolds = 3, num_threads = 1)
model4 <- cv.rbridge(X, y, q = 1, R = R4.mat, r = r4.vec,
                     nlambda = 5, nfolds = 3, num_threads = 1)
# Inspect both selectors and explicitly choose the minimum-error fit.
c(minimum_error = model1$lambda.min, historical_1se = model1$lambda.1se)
coef(model1, s = "lambda.min")
predict(model1, newx = X[1:5, , drop = FALSE], s = "lambda.min")
plot(model1)
# Check the restriction after converting the returned slopes.
x_centered <- sweep(X, 2, colMeans(X), "-")
x_rms <- sqrt(colMeans(x_centered^2))
standardized_slopes <- coef(model1, s = "lambda.min")[-1, 1] * x_rms
as.numeric(R1.mat %*% standardized_slopes - r1.vec)
```

`plot.cv.bridge` *Plot the bridge cross-validation curve*

Description

Display validation mean squared error over the fitted penalty grid, with historical error bars, counts of nonzero slopes and both selected penalties.

Usage

```
## S3 method for class 'cv.bridge'
plot(x, sign.lambda = 1, ...)
```

Arguments

x	Fitted CV object returned by cv.bridge.
sign.lambda	Horizontal-axis direction. The default 1 plots log(lambda); -1 reverses it using -log(lambda). Use positive fitted penalties for the logarithmic axis.
...	Graphical arguments passed to the initial base-R plot() call, such as main, xlab, ylab or ylim.

Details

Points show x$cve; bars span x$cvlo to x$cvup. The bars use the historical cvse calculation described in cv.bridge, rather than conventional foldwise standard errors. The top axis shows full-data nonzero-slope counts, excluding the intercept. Vertical dotted lines mark lambda.min and lambda.1se; the lines coincide when both selectors choose the same fitted penalty. The curve is a diagnostic for the supplied grid and folds; its shape depends on the data and need not be U-shaped.

Value

Draws the plot and returns NULL invisibly.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

cv.bridge, coef.cv.bridge

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.bridge(X, y, q = 1, nlambda = 5,
                    nfolds = 3, num_threads = 1)
plot(model1, main = "Cross-validation over the fitted penalty grid")
# Reverse the log-penalty axis without refitting the model.
plot(model1, sign.lambda = -1)
```

`plot.cv.rbridge` *Plot the restricted bridge cross-validation curve*

Description

Display validation mean squared error over the fitted penalty grid, with historical error bars, counts of nonzero slopes and both selected penalties.

Usage

```
## S3 method for class 'cv.rbridge'
plot(x, sign.lambda = 1, ...)
```

Arguments

`x`	Fitted CV object returned by `cv.rbridge`.
`sign.lambda`	Horizontal-axis direction. The default `1` plots `log(lambda)`; `-1` reverses it using `-log(lambda)`. Use positive fitted penalties for the logarithmic axis.
`...`	Graphical arguments passed to the initial base-R `plot()` call, such as `main`, `xlab`, `ylab` or `ylim`.

Details

Points show `x$cve`; bars span `x$cvlo` to `x$cvup`. The bars use the historical `cvse` calculation described in `cv.rbridge`, rather than conventional foldwise standard errors. The top axis shows full-data nonzero-slope counts, excluding the intercept. Vertical dotted lines mark `lambda.min` and `lambda.1se`; the lines coincide when both selectors choose the same fitted penalty. The curve is a diagnostic for the supplied grid and folds; its shape depends on the data and need not be U-shaped.

Value

Draws the plot and returns `NULL` invisibly.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`cv.rbridge`, `coef.cv.rbridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5,
                     nfolds = 3, num_threads = 1)
plot(model1, main = "Cross-validation over the fitted penalty grid")
# Reverse the log-penalty axis without refitting the model.
plot(model1, sign.lambda = -1)
```

`predict.bridge` *Predict from a fitted bridge object*

Description

Compute response predictions, extract coefficients, or identify nonzero predictors from a fitted bridge object.

Usage

```
## S3 method for class 'bridge'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se"),
  type = c("response", "nonzero", "coefficients"),
  ...
)
```

Arguments

`object`	Fitted object returned by `bridge`, with `p` predictors in its training design.
`newx`	Numeric matrix with one row per new observation and the same `p` columns, in the same order, as the training design. A numeric vector of length `p` represents one observation. Values are supplied on the original predictor scale. Required only for `type = "response"`.

s	Numeric fitted penalty value(s). Values must match `object$lambda` exactly; no interpolation is used. Omitting this argument returns all fitted penalties. The character selector `lambda.min` requests the smallest fitted penalty. Use a CV object for the selector `lambda.1se`.
type	Output to return: `"response"` for fitted response values, `"coefficients"` for intercept and slopes, or `"nonzero"` for predictor indices over the entire fitted path.
...	Additional arguments accepted for compatibility.

Details

Prediction uses the fitted intercept and original-scale slopes. Columns of `newx` are matched by position, not reordered by their names. For `type = "nonzero"`, all fitted penalties are inspected and `s` does not restrict the returned indices.

Value

For `type = "response"`, a numeric vector for one selected penalty or a matrix with one row per new observation and one column per selected penalty. For `type = "coefficients"`, a numeric matrix with `p + 1` rows, including the intercept, even for one penalty. For `type = "nonzero"`, a pathwise collection of 1-based predictor indices excluding the intercept. Its historical container can simplify depending on the support pattern and number of predictors.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

coef.bridge, bridge

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
model1 <- bridge(X, y, q = 1, nlambda = 5)
# Predict five observations at one explicitly selected penalty.
predict(model1, newx = X[1:5, , drop = FALSE], s = model1$lambda[2])
# Extract coefficients using the complete type name.
predict(model1, s = model1$lambda[2], type = "coefficients")
# Predictor indices for every fitted penalty; the intercept is excluded.
predict(model1, type = "nonzero")
```

predict.cv.bridge *Predict from a fitted cv.bridge object*

Description

Compute response predictions, extract coefficients, or identify nonzero predictors from a fitted cv.bridge object.

Usage

```
## S3 method for class 'cv.bridge'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se"),
  type = c("response", "nonzero", "coefficients"),
  ...
)
```

Arguments

object	Fitted object returned by cv.bridge, with p predictors in its training design.
newx	Numeric matrix with one row per new observation and the same p columns, in the same order, as the training design. A numeric vector of length p represents one observation. Values are supplied on the original predictor scale. Required only for type = "response".
s	Penalty selection, using numeric fitted values or a character name. Numeric values must match object$lambda exactly; no interpolation is used. The character selectors are: "lambda.min" Select the penalty with minimum CV error. "lambda.1se" Select using the historical CV error bars. The default is lambda.min.
type	Output to return: "response" for fitted response values, "coefficients" for intercept and slopes, or "nonzero" for predictor indices over the entire fitted path.
...	Additional arguments accepted for compatibility.

Details

Prediction uses the fitted intercept and original-scale slopes. Columns of newx are matched by position, not reordered by their names. For type = "nonzero", all fitted penalties are inspected and s does not restrict the returned indices. The defaults differ between coef() and predict(): coefficient extraction uses lambda.1se, whereas response prediction uses lambda.min. Specify the same s in both calls when comparing coefficients and their predictions.

Value

For `type = "response"`, a numeric vector for one selected penalty or a matrix with one row per new observation and one column per selected penalty. For `type = "coefficients"`, a numeric matrix with `p + 1` rows, including the intercept, even for one penalty. For `type = "nonzero"`, a pathwise collection of 1-based predictor indices excluding the intercept. Its historical container can simplify depending on the support pattern and number of predictors.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`coef.cv.bridge`, `cv.bridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.bridge(X, y, q = 1, nlambda = 5,
                    nfolds = 3, num_threads = 1)
# Predict five observations at one explicitly selected penalty.
predict(model1, newx = X[1:5, , drop = FALSE], s = "lambda.min")
# Extract coefficients using the complete type name.
predict(model1, s = "lambda.min", type = "coefficients")
# Predictor indices for every fitted penalty; the intercept is excluded.
predict(model1, type = "nonzero")
```

`predict.cv.rbridge` *Predict from a fitted cv.rbridge object*

Description

Compute response predictions, extract coefficients, or identify nonzero predictors from a fitted cv.rbridge object.

Usage

```
## S3 method for class 'cv.rbridge'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se"),
  type = c("response", "nonzero", "coefficients"),
  ...
)
```

Arguments

object	Fitted object returned by cv.rbridge, with p predictors in its training design.
newx	Numeric matrix with one row per new observation and the same p columns, in the same order, as the training design. A numeric vector of length p represents one observation. Values are supplied on the original predictor scale. Required only for type = "response".
s	Penalty selection, using numeric fitted values or a character name. Numeric values must match object$lambda exactly; no interpolation is used. The character selectors are: "lambda.min" Select the penalty with minimum CV error. "lambda.1se" Select using the historical CV error bars. The default is lambda.min.
type	Output to return: "response" for fitted response values, "coefficients" for intercept and slopes, or "nonzero" for predictor indices over the entire fitted path.
...	Additional arguments accepted for compatibility.

Details

Prediction uses the fitted intercept and original-scale slopes. Columns of newx are matched by position, not reordered by their names. For type = "nonzero", all fitted penalties are inspected and s does not restrict the returned indices. The defaults differ between coef() and predict(): coefficient extraction uses lambda.1se, whereas response prediction uses lambda.min. Specify the same s in both calls when comparing coefficients and their predictions. Returned slopes are on the original predictor scale; the linear restrictions concern standardized slopes. See rbridge for converting scales before checking restriction residuals.

Value

For type = "response", a numeric vector for one selected penalty or a matrix with one row per new observation and one column per selected penalty. For type = "coefficients", a numeric matrix with p + 1 rows, including the intercept, even for one penalty. For type = "nonzero", a pathwise collection of 1-based predictor indices excluding the intercept. Its historical container can simplify depending on the support pattern and number of predictors.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

coef.cv.rbridge, cv.rbridge

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- cv.rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5,
                     nfolds = 3, num_threads = 1)
# Predict five observations at one explicitly selected penalty.
predict(model1, newx = X[1:5, , drop = FALSE], s = "lambda.min")
# Extract coefficients using the complete type name.
predict(model1, s = "lambda.min", type = "coefficients")
# Predictor indices for every fitted penalty; the intercept is excluded.
predict(model1, type = "nonzero")
```

`predict.rbridge` *Predict from a fitted rbridge object*

Description

Compute response predictions, extract coefficients, or identify nonzero predictors from a fitted rbridge object.

Usage

```
## S3 method for class 'rbridge'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se"),
  type = c("response", "nonzero", "coefficients"),
```

```
  ...
)
```

Arguments

`object`	Fitted object returned by `rbridge`, with `p` predictors in its training design.
`newx`	Numeric matrix with one row per new observation and the same `p` columns, in the same order, as the training design. A numeric vector of length `p` represents one observation. Values are supplied on the original predictor scale. Required only for `type = "response"`.
`s`	Numeric fitted penalty value(s). Values must match `object$lambda` exactly; no interpolation is used. Omitting this argument returns all fitted penalties. The character selector `lambda.min` requests the smallest fitted penalty. Use a CV object for the selector `lambda.1se`.
`type`	Output to return: `"response"` for fitted response values, `"coefficients"` for intercept and slopes, or `"nonzero"` for predictor indices over the entire fitted path.
`...`	Additional arguments accepted for compatibility.

Details

Prediction uses the fitted intercept and original-scale slopes. Columns of `newx` are matched by position, not reordered by their names. For `type = "nonzero"`, all fitted penalties are inspected and `s` does not restrict the returned indices. Returned slopes are on the original predictor scale; the linear restrictions concern standardized slopes. See `rbridge` for converting scales before checking restriction residuals.

Value

For `type = "response"`, a numeric vector for one selected penalty or a matrix with one row per new observation and one column per selected penalty. For `type = "coefficients"`, a numeric matrix with `p + 1` rows, including the intercept, even for one penalty. For `type = "nonzero"`, a pathwise collection of 1-based predictor indices excluding the intercept. Its historical container can simplify depending on the support pattern and number of predictors.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

See Also

`coef.rbridge`, `rbridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5)
# Predict five observations at one explicitly selected penalty.
predict(model1, newx = X[1:5, , drop = FALSE], s = model1$lambda[2])
# Extract coefficients using the complete type name.
predict(model1, s = model1$lambda[2], type = "coefficients")
# Predictor indices for every fitted penalty; the intercept is excluded.
predict(model1, type = "nonzero")
```

`rbridge` *Fit a restricted bridge coefficient path*

Description

Fit a bridge-penalized least-squares path subject to user-supplied linear restrictions on standardized slopes, using local quadratic approximation.

Usage

```
rbridge(
  X,
  y,
  q = 1,
  R,
  r,
  lambda.min = ifelse(n > p, 0.001, 0.05),
  nlambda = 100,
  lambda,
  eta = 1e-07,
  converge = 10^10
)
```

Arguments

X	Numeric design matrix with n observation rows and p predictor columns. Supply finite values and predictors with nonzero variation; omit an intercept column. A data frame is converted using model.matrix(~0+., X); restrictions and future predictions must use the resulting predictor order.
y	Numeric response vector of length n, containing finite values.
q	Bridge-penalty exponent, a finite number in (0, 2]. Values 1 and 2 specify the L1 and L2 penalty forms. Estimation uses local quadratic approximation (LQA).
R	Finite numeric matrix with m linearly independent restriction rows and p predictor columns. Each row specifies a linear combination of the standardized predictor coefficients, in the order of X. The intercept is excluded.
r	Finite numeric target vector of length m, or an m by 1 matrix. The targets have the units of the response when R is dimensionless. See Details for the coefficient scale and a residual check.
lambda.min	Positive absolute endpoint of the automatically generated penalty grid. The default is 0.001 when n > p, and 0.05 otherwise. This argument is a penalty value, not a ratio. In CV results, the field also named lambda.min instead denotes the selected penalty.
nlambda	Number of automatically generated penalty values, an integer of at least 2. The default is 100; ignored when lambda is supplied.
lambda	Optional nonempty numeric vector of finite, nonnegative penalty values. Its order is retained. When supplied, it determines the fitted path and overrides nlambda and the grid argument lambda.min.
eta	Positive numerical tolerance, with default 1e-7. Used for coefficient elimination and stopping on squared coefficient changes.
converge	Nonnegative initial convergence-distance value, retained for compatibility. The default is 1e10. Iteration stops when the updated squared coefficient change is at most eta, or at the historical cap.

Details

Predictors are centered and divided by their centered root-mean-square (RMS) deviations, using the observation count n as the divisor. The response is centered. The intercept is reconstructed after fitting, and returned coefficients are expressed on the original predictor scale. The generated penalties are logarithmically spaced between an internally computed starting value and the absolute endpoint lambda.min. Restrictions act on standardized slopes. To check a fitted original-scale slope vector, multiply it elementwise by the centered RMS deviations of the predictors before applying R. The resulting restriction residual is the fitted linear combination minus r; it should be examined with a numerical tolerance, particularly after coefficient thresholding. For example, a zero restriction on one slope has the same meaning on either scale, whereas a nonzero sum or contrast depends on the scales.

Let Z be the n by p centered, RMS-standardized design, y_c the centered response vector of length n, and b the p-dimensional standardized slope vector. For exponent $q \in (0, 2]$ and penalty weight

$\lambda \geq 0$, Equation (1) defines the penalized least-squares target:

$$\min_{b:\, Rb=r} \left\{ \|y_c - Zb\|_2^2 + \lambda \sum_{j=1}^{p} |b_j|^q \right\}. \qquad (1)$$

In Equation (1), $\|\cdot\|_2^2$ is the squared Euclidean norm, so its first term is the unnormalized residual sum of squares. The index j runs over the predictors; the second term is the bridge penalty, with λ controlling its weight relative to residual error. Here R is an m by p matrix with independent rows and r is the m-dimensional target vector. The condition $Rb = r$ restricts the standardized slopes and excludes the intercept. The fitting routine applies the article's LQA updates, coefficient elimination and stopping rules. The criterion is nonconvex when $q < 1$, so the iterative result is not a guarantee of a global minimum. The getting-started vignette explains the penalty geometry and the fixed-LQA restriction correction.

Value

A list of class `rbridge` with components:

`betas`	Numeric coefficient matrix with `p + 1` rows and `length(lambda)` columns, one column per fitted penalty. The first row is `(Intercept)`; the remaining rows are original-scale slopes in predictor order. Column names identify the fitted penalties.
`lambda`	Numeric vector of penalties, in fitted column order.

Author(s)

Bahadir Yuzbasi
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

References

Yuzbasi, B., Arashi, M. and Akdeniz, F. (2021). Penalized regression via the restricted bridge estimator. Soft Computing, 25, 8401–8416. doi:10.1007/s00500021057639.

See Also

`cv.rbridge`, `coef.rbridge`, `predict.rbridge`

Examples

```
# Reproducible synthetic demonstration: 100 observations, eight predictors.
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
n <- 100
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
y <- as.numeric(X %*% beta + rnorm(n))
# A user-specified sum of standardized slopes: b[1] + b[2] + b[5] = 6.5.
# This target is illustrative; it is not asserted to equal the generating sum.
```

```
R1.mat <- matrix(c(1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r1.vec <- matrix(6.5, nrow = 1, ncol = 1)
# A second standardized-slope contrast: -b[1] + b[2] + b[5] = 0.5.
R2.mat <- matrix(c(-1, 1, 0, 0, 1, 0, 0, 0), nrow = 1)
r2.vec <- matrix(0.5, nrow = 1, ncol = 1)
# Impose both independent restrictions together.
R3.mat <- rbind(R1.mat, R2.mat)
r3.vec <- matrix(c(6.5, 0.5), ncol = 1)
# Alternatively, set the five inactive generating slopes to zero.
p.active <- which(beta != 0)
R4.mat <- diag(p)[-p.active, , drop = FALSE]
r4.vec <- matrix(rep(0, p - length(p.active)), ncol = 1)

# Five penalties keep this help example fast; the default is 100.
model1 <- rbridge(X, y, q = 1, R = R1.mat, r = r1.vec, nlambda = 5)
model2 <- rbridge(X, y, q = 1, R = R2.mat, r = r2.vec, nlambda = 5)
model3 <- rbridge(X, y, q = 1, R = R3.mat, r = r3.vec, nlambda = 5)
model4 <- rbridge(X, y, q = 1, R = R4.mat, r = r4.vec, nlambda = 5)
# Rows include the intercept; columns correspond to fitted penalties.
dim(model1$betas)
# Extract the second fitted penalty exactly; no interpolation is performed.
selected <- model1$lambda[2]
coefficients <- coef(model1, s = selected)
coefficients
predict(model1, newx = X[1:5, , drop = FALSE], s = selected)
# Convert returned slopes to standardized slopes before checking the target.
x_centered <- sweep(X, 2, colMeans(X), "-")
x_rms <- sqrt(colMeans(x_centered^2))
standardized_slopes <- coefficients[-1, 1] * x_rms
as.numeric(R1.mat %*% standardized_slopes - r1.vec)
```

`rbridge_info` *Compiled engine capabilities*

Description

Report whether this installed package was compiled with OpenMP support. This reports a build capability, not a measured speedup or the number of threads used in a particular fit. CV uses at most one worker per fold.

Usage

```
rbridge_info()
```

Details

Without OpenMP, CV runs sequentially for any `num_threads`. With OpenMP, `num_threads` requests fold-level workers. Set BLAS threads to one before starting R to avoid nested parallelism.

Value

A list with three components:

`version`	Installed package version, as a character string.
`openmp`	Logical flag indicating compiled OpenMP support.
`openmp_version`	OpenMP specification date reported by the compiler, or zero when support is absent.

Examples

```
rbridge_info()
```

Index

bridge, 2, *5*, *12*, *19, 20*

coef.bridge, *4*, 5, *20*
coef.cv.bridge, 6, *12*, *17*, *22*
coef.cv.rbridge, 7, *15*, *18*, *24*
coef.rbridge, 9, *25*, *28*
cv.bridge, *4*, *6, 7*, 10, *17*, *21, 22*
cv.rbridge, *7, 8*, 13, *18*, *23, 24*, *28*

plot.cv.bridge, *12*, 16
plot.cv.rbridge, *15*, 18
predict.bridge, *4, 5*, 19
predict.cv.bridge, *7*, *12*, 21
predict.cv.rbridge, *8*, *15*, 22
predict.rbridge, *9*, 24, *28*

rbridge, *8, 9*, *15*, *23*, *25*, 26
rbridge_info, 29