

# Package ‘rbridge’

February 29, 2020

**Type** Package

**Title** Restricted Bridge Estimation

**Version** 1.0.2

**Date** 2020-02-29

**Author** Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz

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

**Description** Bridge Regression estimation with linear restrictions defined in Yuzbasi et al. (2019) <arXiv:1910.03660>. Special cases of this approach fit the restricted LASSO, restricted RIDGE and restricted Elastic Net estimators.

**License** GPL-3

**Imports** Rcpp, Matrix, dplyr, methods

**Suggests** utils, stats, testthat

**Encoding** UTF-8

**LinkingTo** Rcpp, RcppArmadillo

**RoxygenNote** 6.1.1

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2020-02-29 11:40:03 UTC

## R topics documented:

bridge . . . . .	2
coef.bridge . . . . .	3
coef.cv.bridge . . . . .	4
coef.cv.rbridge . . . . .	5
coef.rbridge . . . . .	6
cv.bridge . . . . .	7
cv.rbridge . . . . .	9
plot.cv.bridge . . . . .	11
plot.cv.rbridge . . . . .	12
predict.bridge . . . . .	13

predict.cv.bridge . . . . .	14
predict.cv.rbridge . . . . .	15
predict.rbridge . . . . .	16
rbridge . . . . .	17

<b>Index</b>	<b>20</b>
--------------	-----------

---

bridge	<i>Fit a Bridge Estimation</i>
--------	--------------------------------

---

## Description

Fit a bridge penalized maximum likelihood. It is computed the regularization path which is consisted of lasso or ridge penalty at the a grid values for lambda

## 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	Design matrix.
y	Response vector.
q	is the degree of norm which includes ridge regression with q=2 and lasso estimates with q=1 as special cases
lambda.min	The smallest value for lambda if n>p is 0.001 and 0.05 otherwise.
nlambda	The number of lambda values - default is 100
lambda	A user supplied lambda sequence. By default, the program compute a squence of values the length of nlambda.
eta	is a preselected small positive threshold value. It is deleted jth variable to make the algorithm stable and also is excluded jth variable from the final model. Default is 1e-07.
converge	is the value of converge. Defaults is 10^10. In each iteration, it is calculated by sum of square the change in linear predictor for each coefficient. The algorithm iterates until converge > eta.

## Details

Computes bridge estimation

## Value

An object of class rbridge, a list with entries

betas	Coefficients computed over the path of lambda
lambda	The lambda values which is given at the function

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[cv.bridge](#)

**Examples**

```
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),n,p)
y = X%%beta + rnorm(n)

model1 <- bridge(X, y, q = 1)
print(model1)

model2 <- bridge(X, y, q = 2)
print(model2)
```

---

coef.bridge

*Extract coefficients from a 'bridge' object*

---

**Description**

Extract coefficients from a 'bridge' object.

**Usage**

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

**Arguments**

object	A 'bridge' object.
s	Value(s) of the penalty parameter lambda at which predictions are required.
...	Additional arguments for compatibility.

**Value**

A vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[predict.bridge](#)

**Examples**

```
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),n,p)
y = X%%beta + rnorm(n)

model1 <- bridge(X, y, q = 1)
coef(model1,s='lambda.min')
```

---

coef.cv.bridge

*Extract coefficients from a 'cv.bridge' object*

---

**Description**

Extract coefficients from a 'cv.bridge' object.

**Usage**

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

**Arguments**

object	A 'cv.bridge' object.
s	Value(s) of the penalty parameter lambda at which predictions are required.
...	Additional arguments for compatibility.

**Value**

A vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**[predict.cv.rbridge](#)**Examples**

```
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),n,p)
y = X%%beta + rnorm(n)

model1 <- cv.bridge(X, y, q = 1)
coef(model1,s='lambda.min')
```

---

coef.cv.rbridge	<i>Extract coefficients from a 'cv.rbridge' object</i>
-----------------	--

---

**Description**

Extract coefficients from a 'cv.rbridge' object.

**Usage**

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

**Arguments**

object	A 'cv.rbridge' object.
s	Value(s) of the penalty parameter lambda at which predictions are required.
...	Additional arguments for compatibility.

**Value**

A vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**[predict.cv.rbridge](#)

**Examples**

```

set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)

### Restricted Matrix and vector
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- cv.rbridge(X, y, q = 1, R1.mat, r1.vec)
coef(model1,s='lambda.min')

```

---

coef.rbridge

*Extract coefficients from a 'rbridge' object*


---

**Description**

Makes predictions from a cross-validated 'rbridge' model

**Usage**

```

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

```

**Arguments**

object	A 'rbridge' object.
s	Value(s) of the penalty parameter lambda at which predictions are required.
...	Additional arguments for compatibility.

**Value**

Among a matrix with predictions, a vector non-zero indexing or a vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[predict.rbridge](#)

**Examples**

```

set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)

### Restricted Matrix and vector
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- rbridge(X, y, q = 1, R1.mat, r1.vec)
coef(model1,s='lambda.min')

```

---

cv.bridge

*Cross-validation for bridge*


---

**Description**

Does k-fold cross-validation for bridge, produces a plot, and returns a value for lambda

**Usage**

```

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

```

**Arguments**

X	X matrix as in bridge.
y	response y as in bridge.
q	is the degree of norm which includes ridge regression with q=2 and lasso estimates with q=1 as special cases
lambda	lambda sequence; default is NULL. It is given by user or cv.rbridge chooses its own sequence.
nfolds	number of folds - default is 10.
lambda.min	The smallest value for lambda if n>p is 0.001 and 0.05 otherwise.
nlambdas	The number of lambda values - default is 100
eta	is a preselected small positive threshold value. It is deleted jth variable to make the algorithm stable and also is excluded jth variable from the final model. Default is 1e-07.

converge	is the value of converge. Defaults is $10^{10}$ . In each iteration, it is calculated by sum of square the change in linear predictor for each coefficient. The algorithm iterates until $\text{converge} > \text{eta}$ .
num_threads	Number of threads used for parallel computation over the folds,

**Details**

Computes bridge

**Value**

An object of class rbridge, a list with entries

cve	the mean cross-validated error.
cvse	estimate of standard error of cvm.
cvup	upper curve = $\text{cvm} + \text{cvsd}$ .
cvlo	lower curve = $\text{cvm} - \text{cvsd}$ .
lambda	the values of lambda used in the fits
nz	number of non-zero coefficients at each lambda.
betas	estimated coefficient at each lambda.
lambda.min	value of lambda that gives minimum cve
lambda.1se	largest value of lambda such that error is within 1 standard error of the minimum

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[bridge](#)

**Examples**

```
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),n,p)
y = X%%beta + rnorm(n)

##### Model 1
model1 <- cv.bridge(X, y, q = 1)
print(model1)
coef(model1,s='lambda.min')
predict(model1,newx=X[1:5,], s="lambda.min", type="response")
```



```

predict(model1, s="lambda.min",type="coefficient")

##### Model 2
model2 <- cv.bridge(X, y, q = 2)
print(model2)
coef(model2,s='lambda.min')
predict(model2,newx=X[1:5,], s="lambda.min", type="response")
predict(model2, s="lambda.min",type="coefficient")

```

---

cv.rbridge

*Cross-validation for rbridge*


---

### Description

Does k-fold cross-validation for rbridge, produces a plot, and returns a value for lambda

### 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	X matrix as in rbridge.
y	response y as in rbridge.
q	is the degree of norm which includes ridge regression with q=2 and lasso estimates with q=1 as special cases
R	is m by p (m<p) matrix of constants.
r	is a m-vector of known prespecified constants. If it is given true restriction, then

$$r - R\beta = 0.$$

Values for r should be given as a matrix. See "Examples".

lambda	lambda sequence; default is NULL. It is given by user or cv.rbridge chooses its own sequence.
nfolds	number of folds - default is 10.
lambda.min	The smallest value for lambda if n>p is 0.001 and 0.05 otherwise.
nlambda	The number of lambda values - default is 100
eta	is a preselected small positive threshold value. It is deleted jth variable to make the algorithm stable and also is excluded jth variable from the final model. Default is 1e-07.
converge	is the value of converge. Defaults is 10^10. In each iteration, it is calculated by sum of square the change in linear predictor for each coefficient. The algorithm iterates until converge > eta.
num_threads	Number of threads used for parallel computation over the folds,

**Details**

Computes cv.rbridge

**Value**

An object of class rbridge, a list with entries

cve	the mean cross-validated error.
cvse	estimate of standard error of cvm.
cvup	upper curve = cvm+cvsd.
cvlo	lower curve = cvm-cvsd.
lambda	the values of lambda used in the fits
nz	number of non-zero coefficients at each lambda.
betas	estimated coefficient at each lambda.
lambda.min	value of lambda that gives minimum cve
lambda.1se	largest value of lambda such that error is within 1 standard error of the minimum

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[rbridge](#)

**Examples**

```
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
p.active <- which(beta != 0)

### Restricted Matrix and vector
### Res 1
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)
### Res 2
c2 <- c(-1,1,0,0,1,0,0,0)
R2.mat <- matrix(c2,nrow = 1, ncol = p)
r2.vec <- matrix(c(0.5),nrow = 1, ncol = 1)
### Res 3
R3.mat <- t(matrix(c(c1,c2),nrow = p, ncol = 2))
r3.vec <- matrix(c(6.5,0.5),nrow = 2, ncol = 1)
### Res 4
R4.mat = diag(1,p,p)[-p.active,]
```

```

r4.vec <- matrix(rep(0,p-length(p.active)),nrow = p-length(p.active), ncol = 1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%*%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- cv.rbridge(X, y, q = 1, R1.mat, r1.vec)
print(model1)
coef(model1,s='lambda.min')
coef(model1,s='lambda.1se')
predict(model1,newx=X[1:5,], s="lambda.min", type="response")
predict(model1, s="lambda.min",type="coefficient")
predict(model1, s="lambda.1se",type="coefficient")

##### Model 2 based on second restrictions
model2 <- cv.rbridge(X, y, q = 1, R2.mat, r2.vec)
print(model2)
coef(model2,s='lambda.min')
coef(model2,s='lambda.1se')
predict(model2,newx=X[1:5,], s="lambda.min", type="response")
predict(model2, s="lambda.min",type="coefficient")
predict(model2, s="lambda.1se",type="coefficient")

##### Model 3 based on third restrictions
model3 <- cv.rbridge(X, y, q = 1, R3.mat, r3.vec)
print(model3)
coef(model3,s='lambda.min')
coef(model3,s='lambda.1se')
predict(model3,newx=X[1:5,], s="lambda.min", type="response")
predict(model3, s="lambda.min",type="coefficient")
predict(model3, s="lambda.1se",type="coefficient")

##### Model 4 based on fourth restrictions
model4 <- cv.rbridge(X, y, q = 1, R4.mat, r4.vec)
print(model4)
coef(model4,s='lambda.min')
coef(model4,s='lambda.1se')
predict(model4,newx=X[1:5,], s="lambda.min", type="response")
predict(model4, s="lambda.min",type="coefficient")
predict(model4, s="lambda.1se",type="coefficient")

```

---

plot.cv.bridge

*Plot a 'cv.bridge' object function*


---

### Description

Plots the cross-validation curve, and upper and lower standard deviation curves, as a function of the lambda values used.

**Usage**

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

**Arguments**

x	Design matrix.
sign.lambda	Either plot against log(lambda) (default) or its negative if sign.lambda=-1.
...	Other graphical parameters to plot

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

---

plot.cv.rbridge	<i>Plot a 'cv.rbridge' object function</i>
-----------------	--

---

**Description**

Plots the cross-validation curve, and upper and lower standard deviation curves, as a function of the lambda values used.

**Usage**

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

**Arguments**

x	Design matrix.
sign.lambda	Either plot against log(lambda) (default) or its negative if sign.lambda=-1.
...	Other graphical parameters to plot

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

---

predict.bridge	<i>Make predictions from a 'bridge' object</i>
----------------	--

---

## Description

Makes predictions from a cross-validated 'bridge' model

## Usage

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

## Arguments

object	A 'bridge' object.
newx	Matrix of new values for x at which predictions are to be made.
s	Value(s) of the penalty parameter lambda at which predictions are required.
type	It should one of "response", "nonzero" or "coefficients". The "response" is for predicted values, the "nonzero" is for exacting non-zero coefficients and the "coefficients" is for the estimated coefficients.
...	Additional arguments for compatibility.

## Value

Among a matrix with predictions, a vector non-zero indexing or a vector of coefficients

## Author(s)

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

## See Also

[coef.bridge](#)

## Examples

```
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),n,p)  
y = X%%beta + rnorm(n)
```

```

model1 <- bridge(X, y, q = 1)
predict(model1, newx=X[1:5,], s="lambda.min", type="response")
predict(model1, s="lambda.min", type="coefficient")

```

---

predict.cv.bridge      *Make predictions from a 'cv.bridge' object*

---

### Description

Makes predictions from a cross-validated 'cv.bridge' model

### Usage

```

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

```

### Arguments

object	A 'cv.bridge' object.
newx	Matrix of new values for x at which predictions are to be made.
s	Value(s) of the penalty parameter lambda at which predictions are required.
type	It should be one of "response", "nonzero" or "coefficients". The "response" is for predicted values, the "nonzero" is for exacting non-zero coefficients and the "coefficients" is for the estimated coefficients.
...	Additional arguments for compatibility.

### Value

Among a matrix with predictions, a vector non-zero indexing or a vector of coefficients

### Author(s)

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

### See Also

[coef.cv.bridge](#)

**Examples**

```

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),n,p)
y = X%%beta + rnorm(n)

model1 <- cv.bridge(X, y, q = 1)
coef(model1,s='lambda.min')
predict(model1,newx=X[1:5,], s="lambda.min", type="response")
predict(model1, s="lambda.min",type="coefficient")

```

---

predict.cv.rbridge      *Make predictions from a 'cv.rbridge' object*

---

**Description**

Makes predictions from a cross-validated 'cv.rbridge' model

**Usage**

```

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

```

**Arguments**

object	A 'cv.rbridge' object.
newx	Matrix of new values for x at which predictions are to be made.
s	Value(s) of the penalty parameter lambda at which predictions are required.
type	It should one of "response", "nonzero" or "coefficients". The "response" is for predicted values, the "nonzero" is for exacting non-zero coefficients and the "coefficients" is for the estimated coefficients.
...	Additional arguments for compatibility.

**Value**

Among a matrix with predictions, a vector non-zero indexing or a vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>  
 Bahadir Yuzbasi maintainer Baha

**See Also**[coef.cv.rbridge](#)**Examples**

```

set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)

### Restricted Matrix and vector
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- cv.rbridge(X, y, q = 1, R1.mat, r1.vec)
coef(model1,s='lambda.min')
predict(model1,newx=X[1:5,], s="lambda.min", type="response")
predict(model1, s="lambda.min",type="coefficient")

```

---

predict.rbridge	<i>Make predictions from a 'rbridge' object</i>
-----------------	---

---

**Description**

Makes predictions from a cross-validated 'rbridge' model

**Usage**

```

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

```

**Arguments**

object	A 'rbridge' object.
newx	Matrix of new values for x at which predictions are to be made.
s	Value(s) of the penalty parameter lambda at which predictions are required.
type	It should one of "response", "nonzero" or "coefficients". The "response" is for predicted values, the "nonzero" is for exacting non-zero coefficients and the "coefficients" is for the estimated coefficients.
...	Additional arguments for compatibility.



**Value**

Among a matrix with predictions, a vector non-zero indexing or a vector of coefficients

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[coef.cv.bridge](#)

**Examples**

```
set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)

### Restricted Matrix and vector
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- rbridge(X, y, q = 1, R1.mat, r1.vec)
predict(model1,newx=X[1:5,], s="lambda.min", type="response")
predict(model1, s="lambda.min",type="coefficient")
```

---

rbridge

*Fit a Restricted Bridge Estimation*


---

**Description**

Fit a restricted linear model via bridge penalized maximum likelihood. It is computed the regularization path which is consisted of lasso or ridge penalty at the a grid values for lambda

**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	Design matrix.
y	Response vector.
q	is the degree of norm which includes ridge regression with q=2 and lasso estimates with q=1 as special cases
R	is m by p (m<p) matrix of constants.
r	is a m-vector of known prespecified constants. If it is given true restriction, then

$$r - R\beta = 0.$$

Values for r should be given as a matrix. See "Examples".

lambda.min	The smallest value for lambda if n>p is 0.001 and 0.05 otherwise.
nlambda	The number of lambda values - default is 100
lambda	A user supplied lambda sequence. By default, the program compute a squence of values the length of nlambda.
eta	is a preselected small positive threshold value. It is deleted jth variable to make the algorithm stable and also is excluded jth variable from the final model. Default is 1e-07.
converge	is the value of converge. Defaults is 10^10. In each iteration, it is calculated by sum of square the change in linear predictor for each coefficient. The algorithm iterates until converge > eta.

**Details**

In order to couple the bridge estimator with the restriction  $R\beta = r$ , we solve the following optimization problem

$$\min RSSw.r.t ||\beta||_q \text{ and } R\beta = r.$$

**Value**

An object of class rbridge, a list with entries

betas	Coefficients computed over the path of lambda
lambda	The lambda values which is given at the function

**Author(s)**

Bahadir Yuzbasi, Mohammad Arashi and Fikri Akdeniz  
 Maintainer: Bahadir Yuzbasi <b.yzb@hotmail.com>

**See Also**

[cv.rbridge](#)

**Examples**

```

set.seed(2019)
beta <- c(3, 1.5, 0, 0, 2, 0, 0, 0)
p <- length(beta)
beta <- matrix(beta, nrow = p, ncol = 1)
p.active <- which(beta != 0)

### Restricted Matrix and vector
### Res 1
c1 <- c(1,1,0,0,1,0,0,0)
R1.mat <- matrix(c1,nrow = 1, ncol = p)
r1.vec <- as.matrix(c(6.5),1,1)
### Res 2
c2 <- c(-1,1,0,0,1,0,0,0)
R2.mat <- matrix(c2,nrow = 1, ncol = p)
r2.vec <- matrix(c(0.5),nrow = 1, ncol = 1)
### Res 3
R3.mat <- t(matrix(c(c1,c2),nrow = p, ncol = 2))
r3.vec <- matrix(c(6.5,0.5),nrow = 2, ncol = 1)
### Res 4
R4.mat = diag(1,p,p)[-p.active,]
r4.vec <- matrix(rep(0,p-length(p.active)),nrow = p-length(p.active), ncol = 1)

n = 100
X = matrix(rnorm(n*p),n,p)
y = X%%beta + rnorm(n)

##### Model 1 based on first restrictions
model1 <- rbridge(X, y, q = 1, R1.mat, r1.vec)
print(model1)

##### Model 2 based on second restrictions
model2 <- rbridge(X, y, q = 1, R2.mat, r2.vec)
print(model2)

##### Model 3 based on third restrictions
model3 <- rbridge(X, y, q = 1, R3.mat, r3.vec)
print(model3)

##### Model 4 based on fourth restrictions
model4 <- rbridge(X, y, q = 1, R4.mat, r4.vec)
print(model4)

```

# Index

bridge, [2](#), [8](#)

coef.bridge, [3](#), [13](#)

coef.cv.bridge, [4](#), [14](#), [17](#)

coef.cv.rbridge, [5](#), [16](#)

coef.rbridge, [6](#)

cv.bridge, [3](#), [7](#)

cv.rbridge, [9](#), [18](#)

plot.cv.bridge, [11](#)

plot.cv.rbridge, [12](#)

predict.bridge, [4](#), [13](#)

predict.cv.bridge, [14](#)

predict.cv.rbridge, [5](#), [15](#)

predict.rbridge, [6](#), [16](#)

rbridge, [10](#), [17](#)