

# Package ‘regnet’

June 8, 2019

**Type** Package

**Title** Network-Based Regularization for Generalized Linear Models

**Version** 0.4.0

**Author** Jie Ren, Luann C. Jung, Yinhao Du, Cen Wu, Yu Jiang, Junhao Liu

**Maintainer** Jie Ren <jieren@ksu.edu>

**Description** Network-based regularization has achieved success in variable selection for high-dimensional biological data due to its ability to incorporate correlations among genomic features. This package provides procedures of network-based variable selection for generalized linear models (Ren et al. (2017) <doi:10.1186/s12863-017-0495-5> and Ren et al. (2019) <doi:10.1002/gepi.22194>). Two recent additions are the robust network regularization for the survival response and the network regularization for continuous response. Functions for other regularization methods will be included in the forthcoming upgraded versions.

**Depends** R (>= 3.5.0)

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**Imports** glmnet, stats, Rcpp

**URL** <https://github.com/jrhub/regnet>

**BugReports** <https://github.com/jrhub/regnet/issues>

**RoxygenNote** 6.1.1

**LinkingTo** Rcpp, RcppArmadillo

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2019-06-08 14:30:02 UTC

## R topics documented:

regnet-package . . . . .	2
cv.regnet . . . . .	3
print.cv.regnet . . . . .	6
regnet . . . . .	6
rgn . . . . .	8

<b>Index</b>	<b>10</b>
--------------	-----------

---

regnet-package	<i>Network-Based Regularization for Generalized Linear Models</i>
----------------	---

---

### Description

This package provides implementation of the network-based variable selection method proposed in Ren et al (2017) and the robust network-based method for survival response in Ren et al (2019). In addition to network penalty, regnet also allows users to use classical MCP or LASSO penalty.

### Details

The easy-to-use, integrated interfaces `cv.regnet()` and `regnet()` allow users to flexibly choose the fitting methods they prefer. There are three arguments control the fitting method

- response: three types of response are supported: "binary", "continuous" and "survival".
- penalty: three choices of the penalty functions are available: "network", "mcp" and "lasso".
- robust: whether to use robust methods for modeling (Robust methods are only available for survival response for now).

In penalized regression, the tuning parameter  $\lambda_1$  controls the sparsity of the coefficient profile. For network-based methods, an additional tuning parameter  $\lambda_2$  is needed for controlling the smoothness among coefficient profiles. Typical usage of the package is to have the `cv.regnet()` compute the optimal values of lambdas, then provide them to the `regnet()` function for estimating the coefficients

If the users want to include clinical variables that are not subject to penalty in the model, the argument 'clv' can be used to indicate the positions of clinical variables in the X matrix. e.g. 'clv=(1:5)' meaning that the first five variables in X will not be penalized. It is recommended to put the clinical variables at the beginning of the X matrix in a contiguous way (see the 'Value' section of `regnet()` function). However, non-contiguous indices, e.g. 'clv=(2,4,6)', are also allowed.

### References

Ren, J., Du, Y., Li, S., Ma, S., Jiang, Y. and Wu, C. (2019). Robust network-based regularization and variable selection for high dimensional genomics data in cancer prognosis. *Genet. Epidemiol.*, 43:276-291

Ren, J., He, T., Li, Y., Liu, S., Du, Y., Jiang, Y., and Wu, C. (2017). Network-based regularization for high dimensional SNP data in the case-control study of Type 2 diabetes. *BMC Genetics*, 18(1):44

Wu, C., Jiang, Y., Ren, J., Cui, Y., Ma, S. (2018). Dissecting gene-environment interactions: A penalized robust approach accounting for hierarchical structures. *Statistics in Medicine*, 37:437–456

Wu, C., and Ma, S. (2015). A selective review of robust variable selection with applications in bioinformatics. *Briefings in Bioinformatics*, 16(5), 873–883

Wu, C., Shi, X., Cui, Y. and Ma, S. (2015). A penalized robust semiparametric approach for gene-environment interactions. *Statistics in Medicine*, 34 (30): 4016–4030

## See Also

[cv.regnet](#) [regnet](#)

## Examples

```
## Survival response using robust network method
data(SurvExample)
X = rgn.surv$X
Y = rgn.surv$Y
clv = c(1:5) # variables 1 to 5 are treated as clinical variables, we choose not to penalize them.
out = cv.regnet(X, Y, response="survival", penalty="network", clv=clv, robust=TRUE, verbo = TRUE)
out$lambda

b = regnet(X, Y, "survival", "network", out$lambda[1,1], out$lambda[1,2], clv=clv, robust=TRUE)
index = which(rgn.surv$beta[-(1:6)] != 0) # [-(1:6)] removes the intercept and clinical variables
pos = which(b[-(1:6)] != 0)
tp = length(intersect(index, pos))
fp = length(pos) - tp
list(tp=tp, fp=fp)
```

---

cv.regnet

*k-folds cross-validation for regnet*

---

## Description

This function does k-fold cross-validation for regnet and returns the optimal value(s) of lambda.

## Usage

```
cv.regnet(X, Y, response = c("binary", "continuous", "survival"),
  penalty = c("network", "mcp", "lasso"), lamb.1 = NULL,
  lamb.2 = NULL, folds = 5, r = NULL, clv = NULL,
  initiation = NULL, alpha.i = 1, robust = FALSE, verbo = FALSE)
```

**Arguments**

X	X matrix as in regnet.
Y	response Y as in regnet.
response	response type. regnet now supports three types of response: "binary", "continuous" and "survival".
penalty	penalty type. regnet provides three choices for the penalty function: "network", "mcp" and "lasso".
lamb.1	a user-supplied sequence of $\lambda_1$ values, which serves as a tuning parameter to impose sparsity. If it is left as NULL, regnet will compute its own sequence.
lamb.2	a user-supplied sequence of $\lambda_2$ values for network method. $\lambda_2$ controls the smoothness among coefficient profiles. If it is left as NULL, a default sequence will be used.
folds	the number of folds for cross-validation; default is 5.
r	the regularization parameter in MCP; default is 5. For binary response, r should be larger than 4.
clv	a value or a vector, indexing variables that are not subject to penalty. clv only works for continuous and survival responses for now, and will be ignored for other types of responses.
initiation	the method for initiating the coefficient vector. For binary and continuous response, the default is elastic-net, and for survival response the default is zero.
alpha.i	the elastic-net mixing parameter. The program can use the elastic-net for choosing initial values of the coefficient vector. alpha.i is the elastic-net mixing parameter, with $0 \leq \text{alpha.i} \leq 1$ . alpha.i=1 is the lasso penalty, and alpha.i=0 is the ridge penalty. If the user chooses a method other than elastic-net for initializing coefficients, alpha.i will be ignored.
robust	logical flag. Whether or not to use robust methods. Robust methods are only available for survival response in the current version of regnet.
verbo	output progress to the console.

**Details**

When lamb.1 is left as NULL, regnet computes its own sequence. You can find the lamb.1 sequence used by the program in the returned CVM matrix (see the 'Value' section). If you find the default sequence does not work well, you can try (1) standardize the response vector Y; or (2) provide a customized lamb.1 sequence for your data.

Sometimes multiple optimal values(pairs) of lambda(s) can be found (see 'Value'). This is usually normal when the response is binary. However, if the response is survival or continuous, you may want to check (1) if the sequence of lambda is too large (i.e. all coefficients are shrunken to zero under all values of lambda) ; or (2) if the sequence is too small (i.e. all coefficients are non-zero under all values of lambda). If neither, simply choose the value(pair) of lambda based on your preference.

**Value**

an object of class "cv.regnet" is returned, which is a list with components:

lambda	the optimal value(s) of $\lambda$ . More than one values will be returned, if multiple lambdas have the cross-validated error = min(cross-validated errors). If the network penalty is used, lambda contains optimal pair(s) of $\lambda_1$ and $\lambda_2$ .
mcvm	the cross-validated error of the optimal $\lambda$ . For binary response, the error is misclassification rate. For continuous response, mean squared error (MSE) is used. For survival response, the MSE is used for non-robust methods, and the criterion for robust methods is least absolute deviation (LAD).
CVM	a matrix of the mean cross-validated errors of all lambdas used in the fits. The row names of CVM are the values of $\lambda_1$ . If the network penalty was used, the column names are the values of $\lambda_2$ .

**References**

Ren, J., Du, Y., Li, S., Ma, S., Jiang, Y. and Wu, C. (2019). Robust network-based regularization and variable selection for high dimensional genomics data in cancer prognosis. *Genet. Epidemiol.*, 43:276-291

Ren, J., He, T., Li, Y., Liu, S., Du, Y., Jiang, Y., and Wu, C. (2017). Network-based regularization for high dimensional SNP data in the case-control study of Type 2 diabetes. *BMC Genetics*, 18(1):44

**See Also**

[regnet](#)

**Examples**

```
## Binary response using network method
data(LogisticExample)
X = rgn.logi$X
Y = rgn.logi$Y
out = cv.regnet(X, Y, response="binary", penalty="network", folds=5, r = 4.5)
out$lambda
b = regnet(X, Y, "binary", "network", out$lambda[1,1], out$lambda[1,2], r = 4.5)
index = which(rgn.logi$beta != 0)
pos = which(b != 0)
tp = length(intersect(index, pos))
fp = length(pos) - tp
list(tp=tp, fp=fp)

## Binary response using MCP method
out = cv.regnet(X, Y, response="binary", penalty="mcp", folds=5, r = 4.5)
out$lambda
b = regnet(X, Y, "binary", "mcp", out$lambda[1], r = 4.5)
index = which(rgn.logi$beta != 0)
pos = which(b != 0)
tp = length(intersect(index, pos))
fp = length(pos) - tp
```

```
list(tp=tp, fp=fp)
```

---

```
print.cv.regnet      print a cv.glmnet object
```

---

### Description

Print a summary of a cv.glmnet object

### Usage

```
## S3 method for class 'cv.regnet'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

### Arguments

x	cv.glmnet object.
digits	significant digits in printout.
...	other print arguments

### See Also

[cv.regnet](#)

---

```
regnet      fit a regression for given lambda with network-based regularization
```

---

### Description

Network-based penalization regression for given values of  $\lambda_1$  and  $\lambda_2$ . Typical usage is to have the cv.regnet function compute the optimal lambdas, then provide them to the regnet function. Users could also use MCP or Lasso.

### Usage

```
regnet(X, Y, response = c("binary", "continuous", "survival"),
       penalty = c("network", "mcp", "lasso"), lamb.1 = NULL,
       lamb.2 = NULL, r = NULL, clv = NULL, initiation = NULL,
       alpha.i = 1, robust = FALSE)
```

## Arguments

X	matrix of predictors without intercept. Each row should be an observation vector. A column of 1 will be added to the X matrix by the program as the intercept.
Y	response variable. For response="binary", Y should be a numeric vector with zeros and ones. For response="survival", Y should be a two-column matrix with columns named 'time' and 'status'. The latter is a binary variable, with '1' indicating a event, and '0' indicating censoring.
response	response type. regnet now supports three types of response: "binary", "continuous" and "survival".
penalty	penalty type. regnet provides three choices for the penalty function: "network", "mcp" and "lasso".
lamb.1	the tuning parameter $\lambda_1$ that imposes sparsity.
lamb.2	the tuning parameter $\lambda_2$ that controls the smoothness among coefficient profiles. $\lambda_2$ is needed for network penalty.
r	the regularization parameter in MCP. For binary response, r should be larger than 4.
clv	a value or a vector, indexing variables that are not subject to penalty. clv only works for continuous and survival responses for now, and will be ignored for other types of responses.
initiation	method for initiating the coefficient vector. For binary and continuous response, the default is elastic-net, and for survival response the default is zero.
alpha.i	the elastic-net mixing parameter. The program can use the elastic-net for choosing initial values of the coefficient vector. alpha.i is the elastic-net mixing parameter, with $0 \leq \text{alpha.i} \leq 1$ . alpha.i=1 is the lasso penalty, and alpha.i=0 is the ridge penalty. If the user chooses a method other than elastic-net for initializing coefficients, alpha.i will be ignored.
robust	logical flag. Whether or not to use robust methods. Robust methods are only available for survival response.

## Details

The current version of regnet supports two types of responses: "binary", "continuous" and "survival". regnet(..., response="binary", penalty="network") fits a network-based penalized logistic regression; regnet(..., response="continuous", penalty="network") fits a network-based least square regression; regnet(..., response="survival", penalty="network") fits a robust regularized AFT model using network penalty. Please see the references for more details of the models. By default, regnet uses robust methods for survival response. If users would like to use non-robust methods, simply set robust=FALSE. User could also use MCP or Lasso penalty.

The coefficients are always estimated on a standardized X matrix. regnet standardizes each columns of X to have unit variance (using  $1/n$  rather than  $1/(n-1)$  formula). If the coefficients on the original scale are needed, the user can refit a standard model using the subset of variables that have non-zero coefficients.

**Value**

a vector of estimated coefficients. Please note that, if there are variables not subject to penalty (indicated by `clv`), the order of returned vector is `c(Intercept, unpenalized coefficients of clv variables, penalized coefficients of other variables)`.

**References**

Ren, J., He, T., Li, Y., Liu, S., Du, Y., Jiang, Y., and Wu, C. (2017). Network-based regularization for high dimensional SNP data in the case-control study of Type 2 diabetes. *BMC Genetics*, 18(1):44

Ren, J., Du, Y., Li, S., Ma, S., Jiang, Y. and Wu, C. (2019). Robust network-based regularization and variable selection for high dimensional genomics data in cancer prognosis. *Genet. Epidemiol.*, 43:276-291

**See Also**

[cv.regnet](#)

**Examples**

```
## Survival response
data(SurvExample)
X = rgn.surv$X
Y = rgn.surv$Y
clv = c(1:5) # variables 1 to 5 are clinical variables which we choose not to penalize.
penalty = "network"
b = regnet(X, Y, "survival", penalty, rgn.surv$lamb1, rgn.surv$lamb2, clv=clv, robust=TRUE)
index = which(rgn.surv$beta != 0)
pos = which(b != 0)
tp = length(intersect(index, pos))
fp = length(pos) - tp
list(tp=tp, fp=fp)
```

---

 rgn

---

*simulated data for demonstrating the features of regnet*


---

**Description**

Simulated data for demonstrating the features of regnet.

**Usage**

```
data("LogisticExample")
data("SurvExample")
```

**Format**

Each data consists of three components: `X`, `Y`, and `beta`; `beta` is a vector of the true coefficients used to generate `Y`.



**Examples**

```
data("LogisticExample")  
lapply(rgn.logi, class)
```

# Index

\*Topic **datasets**

rgn, 8

\*Topic **models**

cv.regnet, 3

regnet, 6

\*Topic **overview**

regnet-package, 2

cv.regnet, 3, 3, 6, 8

print.cv.regnet, 6

regnet, 3, 5, 6

regnet-package, 2

rgn, 8