Package 'RIVER'

November 12, 2026

Title R package for RIVER (RNA-Informed Variant Effect on Regulation)

Description An implementation of a probabilistic modeling framework that jointly analyzes personal genome and transcriptome data to estimate the probability that a variant has regulatory impact in that individual. It is based on a generative model that assumes that genomic annotations, such as the location of a variant with respect to regulatory elements, determine the prior probability that variant is a functional regulatory variant, which is an unobserved variable. The functional regulatory variant status then influences whether nearby genes are likely to display outlier levels of gene expression in that person. See the RIVER website for more information, documentation and examples.

Version 1.36.0

Author Yungil Kim [aut, cre], Alexis Battle [aut]

Maintainer Yungil Kim <ipw012@gmail.com>

URL https://github.com/ipw012/RIVER

BugReports https://github.com/ipw012/RIVER/issues

Depends R (>= 3.3.2)

biocViews GeneExpression, GeneticVariability, SNP, Transcription, FunctionalPrediction, GeneRegulation, GenomicVariation, BiomedicalInformatics, FunctionalGenomics, Genetics, SystemsBiology, Transcriptomics, Bayesian, Clustering, TranscriptomeVariant, Regression

Imports glmnet, pROC, ggplot2, graphics, stats, Biobase, methods, utils

License GPL (>= 2)

RoxygenNote 6.0.1

LazyData true

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, testthat, devtools

git_url https://git.bioconductor.org/packages/RIVER

git_branch RELEASE_3_23

git_last_commit 12ea791

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

appRIVER 2
evaRIVER 3
getData 4
getFuncRvFeat 5
getFuncRvPosteriors 6
integratedEM 7
mleBeta 8
mleTheta 9
plotPosteriors 10
RIVER 11
testPosteriors 12

Index **14**

`appRIVER` *Application of RIVER*

Description

`appRIVER` trains RIVER with all instances and computes posterior probabilities of FR for downstream analyses.

Usage

```
appRIVER(dataInput, pseudoc = 50, theta_init = matrix(c(0.99, 0.01, 0.3,
  0.7), nrow = 2), costs = c(100, 10, 1, 0.1, 0.01, 0.001, 1e-04),
  verbose = FALSE)
```

Arguments

`dataInput`	An object of ExpressionSet class which contains input data required for all functions in RIVER including genomic features, outlier status, and N2 pairs.
`pseudoc`	Pseudo count.
`theta_init`	Initial values of theta.
`costs`	Candidate penalty parameter values for L2-regularized logistic regression.
`verbose`	Logical option for showing extra information on progress.

Value

A list which contains subject IDs, gene names, posterior probabilities from GAM and RIVER, and estimated parameters from RIVER with used hyperparameters.

Warning

To input a vector of candidate penalty values makes `glmnet` faster than to input a single penalty value

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

`cv.glmnet`, `predict`, `integratedEM`, `testPosteriors`, `getData`, `exprs`

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
postprobs <- appRIVER(dataInput, verbose=TRUE)
```

---

evaRIVER *Evaluation of RIVER*

Description

`evaRIVER` trains RIVER by holding out a list of individual and gene pairs having same rare variants for evaluation, computes test posterior probabilities of FR for 1st individual, and compares them with outlier status of 2nd individual from the list.

Usage

```
evaRIVER(dataInput, pseudoc = 50, theta_init = matrix(c(0.99, 0.01, 0.3,
  0.7), nrow = 2), costs = c(100, 10, 1, 0.1, 0.01, 0.001, 1e-04),
  verbose = FALSE)
```

Arguments

`dataInput`	An object of ExpressionSet class which contains input data required for all functions in RIVER including genomic features, outlier status, and N2 pairs.
`pseudoc`	Pseudo count.
`theta_init`	Initial values of theta.
`costs`	Candidate penalty parameter values for L2-regularized logistic regression.
`verbose`	Logical option for showing extra information on progress.

Value

A list which contains two AUC values from RIVER and GAM, computed specificities and sensitivities from two models, and P-value of comparing the two AUC values.

Warning

A vector of candidate penalty values make `glmnet` faster than to input a single penalty value

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

`cv.glmnet`, `predict`, `integratedEM`, `testPosteriors`, `getData`, `exprs`

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
evaROC <- evaRIVER(dataInput, verbose=TRUE)
```

`getData` *Get coordinated data from a compressed file for RIVER*

Description

`getData` extracts genomic features, z-scores of gene expression, and N2 pairs having same rare variants from an imported compressed data, computes outlier status from z-scores given a z-score threshold and coordinates the genomic features, outlier status, and a list of N2 pairs into ExpressionSet class having standardized data structure.

Usage

```
getData(filename = system.file("extdata", "simulation_RIVER.gz", package =
  "RIVER"), ZscoreThrd = 1.5)
```

Arguments

`filename`	A full path of a compressed input file that consists of all samples in rows and subject ID, gene name, genomic features, z-scores of corresponding gene expression, and a list of N2 pairs in columns from left to right. In N2 pairs, samples not paired with othersamples have NA while two samples sharing same rare variant near a gene have same pre-assigend integers.
`ZscoreThrd`	A |Z-score| threshold for defining outlier status of samples

Value

dataInput An object of ExpressionSet class which contains input data required for all functions in RIVER including genomic features, outlier status, and N2 pairs.

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

fread, ExpressionSet,

Examples

```
InputData <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
```

---

getFuncRvFeat *Posterior probabilities of FR given G*

Description

getFuncRvFeat computes posterior probabilities of FR (functionality of regulatory variant) given G (genomic features) and current estimate of beta (parameters between FR and G).

Usage

```
getFuncRvFeat(Feat, logistic.model, lambda)
```

Arguments

Feat	Genomic features (G)
logistic.model	Logistic regression model with current estimate of beta
lambda	Selected lambda

Value

probabilities of FR given genomic features, P(FR | G)

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

predict

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
costs <- c(100, 10, 1, .1, .01, 1e-3, 1e-4)
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha = 0, nfolds=10)
probFuncRvFeat <- getFuncRvFeat(Feat, logistic.model=logisticAllCV$glmnet.fit,
        lambda=logisticAllCV$lambda.min)
```

---

getFuncRvPosteriors *Posterior probabilities of FR given G and E.*

Description

`getFuncRvPosteriors` computes posterior probabilities of functionality of regulatory variant (FR) given genomic features (G) and outlier status (E) with current estimate of beta (parameters between FR and G) and theta (parameters between FR and E).

Usage

```
getFuncRvPosteriors(Out, probFuncRvFeat, theta)
```

Arguments

`Out`	Binary values of outlier status (E).
`probFuncRvFeat`	probabilities of FR given genomic features and estimated beta, P(FR | G, beta), from `getFuncRvFeat`.
`theta`	Current estimate of theta.

Value

posterior probabilities of FR (P(FR | G, E, beta, theta)) and probable status of FR.

Author(s)

Yungil Kim, <ipw012@gmail.com>

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
theta.init<-matrix(c(.99, .01, .3, .7), nrow=2)
costs<-c(100, 10, 1, .1, .01, 1e-3, 1e-4)
```

```
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha=0, nfolds=10)
probFuncRvFeat <- getFuncRvFeat(Feat, logisticAllCV$glmnet.fit,
        logisticAllCV$lambda.min)
posteriors <- getFuncRvPosteriors(Out, probFuncRvFeat, theta=theta.init)
```

`integratedEM` *An iterative expectation-maximization algorithm for RIVER*

Description

`integratedEM` iteratively executes e-step and m-step until it converges. This is a main function of RIVER.

Usage

```
integratedEM(Feat, Out, lambda, logistic.init, pseudoc, theta.init, costs,
  verbose = FALSE)
```

Arguments

`Feat`	Genomic features (G).
`Out`	Binary values of outlier status (E).
`lambda`	Selected lambda.
`logistic.init`	Smart initialization of beta (parameters between FR and G) from estimate of beta with E via multivariate logistic regression.
`pseudoc`	Pseudo count.
`theta.init`	Initial theta (parameters between FR (functionality of regulatory variant) and E).
`costs`	Candidate penalty parameter values for L2-regularization within logistic regression.
`verbose`	Logical option for showing extra information on progress.

Value

Best estimate of beta and theta, final multivariate logistic regression model, and posterior probabilities of FR.

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

`getFuncRvFeat`, `getFuncRvPosteriors`, `mleTheta`, `mleBeta`, `cv.glmnet`, and `https://github.com/ipw012/RIVER`

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
theta.init=matrix(c(.99, .01, .3, .7), nrow=2)
costs <- c(100, 10, 1, .1, .01, 1e-3, 1e-4)
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha = 0, nfolds=10)
emModelAll <- integratedEM(Feat, Out, lambda=logisticAllCV$lambda.min,
        logistic.init=logisticAllCV$glmnet.fit, pseudoc=50, theta=theta.init,
        costs, verbose=FALSE)
```

---

mleBeta *Maximum likelihoood estimate of beta.*

Description

mleBeta computes maximum likelihoood estimate of beta (parameters between FR (functionality of regulatory variant) and G (genomic annotations); multivariate logistic regression).

Usage

```
mleBeta(Feat, FuncRv, costs)
```

Arguments

Feat	Genomic features (G)
FuncRv	Soft-assignments of FR from E-step
costs	Candidate penalty parameter values for L2-regularization within logistic regression.

Value

MLE of beta

Warning

To input a vector of candidate penalty values makes glmnet faster than to input a single penalty value

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

glmnet

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
theta.init <- matrix(c(.99, .01, .3, .7), nrow=2)
costs <- c(100, 10, 1, .1, .01, 1e-3, 1e-4)
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha=0, nfolds=10)
probFuncRvFeat <- getFuncRvFeat(Feat, logisticAllCV$glmnet.fit, logisticAllCV$lambda.min)
posteriors <- getFuncRvPosteriors(Out, probFuncRvFeat, theta=theta.init)
logistic.cur <- mleBeta(Feat, FuncRv=posteriors$posterior, costs)
```

`mleTheta` *Maximum likelihoood estimate of theta.*

Description

`mleTheta` computes maximum likelihoood estimate of theta (parameters between FR (functionality of regulatory variant) and E (outlier status); Naive-Bayes).

Usage

```
mleTheta(Out, FuncRv, pseudocount)
```

Arguments

`Out`	Binary values of outlier status (E).
`FuncRv`	Soft-assignments of FR from E-step
`pseudocount`	Pseudo count.

Value

MLE of theta

Author(s)

Yungil Kim, <ipw012@gmail.com>

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
theta.init <- matrix(c(.99, .01, .3, .7), nrow=2)
costs <- c(100, 10, 1, .1, .01, 1e-3, 1e-4)
```

```
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha = 0, nfolds=10)
probFuncRvFeat <- getFuncRvFeat(Feat, logisticAllCV$glmnet.fit, logisticAllCV$lambda.min)
posteriors <- getFuncRvPosteriors(Out, probFuncRvFeat, theta=theta.init)
thetaCur <- mleTheta(Out, FuncRv=posteriors$posterior, pseudoc=50)
```

`plotPosteriors` *Draw scatter plots of posterior probabilities from both RIVER GAM in terms of outlier status.*

Description

`plotPosteriors` draws scatter plots of posterior probabilities from both RIVER GAM (genomic annotation model) in terms of outlier status.

Usage

```
plotPosteriors(postprobs, outliers)
```

Arguments

`postprobs`	Output of `evaRIVER`, which provides test posterior probabilities from both RIVER and GAM for all instances.
`outliers`	Outlier status of examples

Value

A figure of posteriors from RIVER (y-axis) and GAM (x-axis) models for ouliters and non-outliers separately

Author(s)

Yungil Kim, <ipw012@gmail.com>

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
postprobs <- appRIVER(dataInput)
plotPosteriors(postprobs, outliers=as.numeric(unlist(dataInput$Outlier))-1)
```

RIVER *RIVER: R package for an implementation of an extensible predictive model for combining whole genome sequences with molecular phenotypes to identify high-impact rare variants.*

Description

The RIVER package provides two applicable functions of RIVER to actual datasets including genomic features and outlier status and a list of required functions for RIVER: evalRIVER, appRIVER, getFRPosteriors, mleTheta, mleBeta, getFRgivenG, testPosteriors, and integratedEM.

getFuncRvFeat

The `getFuncRvFeat` computes posterior probabilities of FR (functionality of regulatory variant) given genomic features (G) and current estimate of beta (parameters between FR and G).

getFuncRvPosteriors

The `getFuncRvPosteriors` computes posterior probabilities of FR (functionality of regulatory variant) given genomic features (G) and outlier status (E) with current estimate of beta (parameters between FR and G) and theta (parameters between FR and E).

mleBeta

The `mleBeta` computes maximum likelihoood estimate of beta (parameters between FR and G; multivariate logistic regression).

mleTheta

The `mleTheta` computes maximum likelihoood estimate of theta (parameters between FR and E; Naive-Bayes).

testPosteriors

The `testPosteriors` computes posterior probabilities of FR (functionality of regulatory variant) given G (genomic annotations) and E (outlier status) with estimate of beta (parameters between FR and G) and theta (parameters between FR and E).

integratedEM

The `integratedEM` iteratively executes e-step and m-step until it converges. This is a main function of RIVER.

plotPosteriors

The `plotPosteriors` draw scatter plots of posterior probabilities from both RIVER GAM in terms of outliers status.

evalRIVER

The evaRIVER trains RIVER by holding out a list of individual and gene pairs having same rare variants for evaluation, computes test posterior probabilities of FR (functionality of regulatory variant) for 1st individual, and compares them with outlier status of 2nd individual from the list.

appRIVER

The appRIVER trains RIVER with all instances and computes posterior probabilities of FR (functionality of regulatory variant) for downstream analyses.

Source

https://github.com/ipw012/RIVER

---

testPosteriors *Test posterior probabilities of FR given G and E*

Description

testPosteriors computes posterior probabilities of FR (functionality of regulatory variant) given G (genomic annotations) and E (outlier status) with estimate of beta (parameters between FR and G) and theta (parameters between FR and E).

Usage

```
testPosteriors(Feat, Out, emModel)
```

Arguments

Feat	Genomic features (G)
Out	Binary values of outlier status (E).
emModel	Estimated parameters including beta and theta via EM and selected lambdas

Value

test posterior probabilities of FR given new outlier status (E) and genomic features (G), P(FR | G, E, beta, theta), and probable status of FR.

Author(s)

Yungil Kim, <ipw012@gmail.com>

See Also

getFuncRvFeat and getFuncRvPosteriors

Examples

```
dataInput <- getData(filename=system.file("extdata", "simulation_RIVER.gz",
        package = "RIVER"), ZscoreThrd=1.5)
Feat <- scale(t(Biobase::exprs(dataInput))) # genomic features (G)
Out <- as.vector(as.numeric(unlist(dataInput$Outlier))-1) # outlier status (E)
theta.init <- matrix(c(.99, .01, .3, .7), nrow=2)
costs <- c(100, 10, 1, .1, .01, 1e-3, 1e-4)
logisticAllCV <- glmnet::cv.glmnet(Feat, Out, lambda=costs, family="binomial",
        alpha = 0, nfolds=10)
emModelAll <- integratedEM(Feat, Out, logisticAllCV$lambda.min, logisticAllCV$glmnet.fit,
        pseudoc=50, theta.init, costs, verbose=FALSE)
trainedpost <- testPosteriors(Feat, Out, emModel=emModelAll)
```

Index

appRIVER, 2

cv.glmnet, *3, 4*, *7*

evaRIVER, 3
ExpressionSet, *5*
exprs, *3, 4*

fread, *5*

getData, *3, 4*, 4
getFuncRvFeat, 5, *7*, *12*
getFuncRvPosteriors, 6, *7*, *12*
glmnet, *8*

integratedEM, *3, 4*, 7

mleBeta, *7*, 8
mleTheta, *7*, 9

plotPosteriors, 10
predict, *3–5*

RIVER, 11
RIVER-package *(*RIVER*)*, 11

testPosteriors, *3, 4*, 12