Package ‘selectKSigs’

April 1, 2025

Type Package

Title Selecting the number of mutational signatures using a perplexity-based measure and cross-validation

Depends R(>= 3.6)

Imports HiLDA, magrittr, gtools, methods, Rcpp

Suggests knitr, rmarkdown, testthat, BiocStyle, ggplot2, dplyr, tidyr

Version 1.18.0

Date 2021-10-18

Description A package to suggest the number of mutational signatures in a collection of somatic mutations using calculating the cross-validated perplexity score.

URL `https://github.com/USCbiostats/selectKSigs`

BugReports `https://github.com/USCbiostats/HiLDA/selectKSigs`

License GPL-3

biocViews Software, SomaticMutation, Sequencing, StatisticalMethod, Clustering

RoxygenNote 7.1.2

LinkingTo Rcpp

VignetteBuilder knitr

Encoding UTF-8

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

git_branch RELEASE_3_20

git_last_commit 2cc78e8

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2025-03-31

Author Zhi Yang [aut, cre],
Yuichi Shiraishi [ctb]

Maintainer Zhi Yang `<zyang895@gmail.com>`

Contents

`calcPMSLikelihood` *A function for calculating the log-likelihood from the data and parameters*

Description

A function for calculating the log-likelihood from the data and parameters

Usage

```
calcPMSLikelihood(p, y)
```

Arguments

`p`	this variable includes the parameters for mutation signatures and membership parameters
`y`	this variable includes the information on the mutation features, the number of mutation signatures specified and so on

Value

a value

`Calculate_Likelihood_test` *Output the maximum potential scale reduction statistic of all parameters estimated*

Description

Output the maximum potential scale reduction statistic of all parameters estimated

Usage

```
Calculate_Likelihood_test(train, test, paramG)
```

Arguments

`train`	a MutationFeatureData S4 class output of training data.
`test`	a MutationFeatureData S4 class output of test data.
`paramG`	an estimatedParameters S4 class with estimated parameters

Value

the likelihood of the test data

`convertFromTurbo_F` *Restore the converted parameter F for turboEM*

Description

Restore the converted parameter F for turboEM

Usage

```
convertFromTurbo_F(turboF, fdim, signatureNum, isBackground)
```

Arguments

`turboF`	F (converted for turboEM)
`fdim`	a vector specifying the number of possible values for each mutation signature
`signatureNum`	the number of mutation signatures
`isBackground`	the logical value showing whether a background mutaiton features is included or not

Value

a vector

`convertFromTurbo_Q` *Restore the converted parameter Q for turboEM*

Description

Restore the converted parameter Q for turboEM

Usage

```
convertFromTurbo_Q(turboQ, signatureNum, sampleNum)
```

Arguments

`turboQ`	Q (converted for turboEM)
`signatureNum`	the number of mutation signatures
`sampleNum`	the number of cancer genomes

Value

a vector

`convertToTurbo_F` *Convert the parameter F so that turboEM can treat*

Description

Convert the parameter F so that turboEM can treat

Usage

```
convertToTurbo_F(vF, fdim, signatureNum, isBackground)
```

Arguments

`vF`	F (converted to a vector)
`fdim`	a vector specifying the number of possible values for each mutation signature
`signatureNum`	the number of mutation signatures
`isBackground`	the logical value showing whether a background mutaiton features is included or not

Value

a vector

`convertToTurbo_Q` *Convert the parameter Q so that turboEM can treat*

Description

Convert the parameter Q so that turboEM can treat

Usage

```
convertToTurbo_Q(vQ, signatureNum, sampleNum)
```

Arguments

`vQ`	Q (converted to a vector)
`signatureNum`	the number of mutation signatures
`sampleNum`	the number of cancer genomes

Value

a vector

`cv_PMSignature` *Output the maximum potential scale reduction statistic of all parameters estimated*

Description

Output the maximum potential scale reduction statistic of all parameters estimated

Usage

```
cv_PMSignature(inputG, Kfold = 3, nRep = 3, Klimit = 8)
```

Arguments

`inputG`	a MutationFeatureData S4 class.
`Kfold`	an integer number of the number of cross-validation folds.
`nRep`	an integer number of replications.
`Klimit`	an integer of the maximum value of number of signatures.

Value

a matrix of measures

Examples

```
load(system.file("extdata/sample.rdata", package = "selectKSigs"))
results <- cv_PMSignature(G, Kfold = 3)
```

`getBG` *Get the statsus of using the background signature*

Description

Get the statsus of using the background signature

Usage

```
getBG(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

the status of using the background signature

`getCounts` *Get the count data in a matrix*

Description

Get the count data in a matrix

Usage

```
getCounts(object)
```

Arguments

`object`	the MutationFeatureData class

Value

the count data in a matrix

`getExposures` *Get a matrix of mutational exposures of signatures*

Description

Get a matrix of mutational exposures of signatures

Usage

```
getExposures(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

a matrix of mutational exposures of signatures

`getFeatures` *Get a vector of possible features*

Description

Get a vector of possible features

Usage

```
getFeatures(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

a vector of possible features

`getFeatureVec` *Get a matrix of feature vector list*

Description

Get a matrix of feature vector list

Usage

```
getFeatureVec(object)
```

Arguments

`object`	the MutationFeatureData class

Value

a matrix of feature vector list

`getK` *Get the number of signatures*

Description

Get the number of signatures

Usage

```
getK(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

the number of signatures in `pmgetSignature` in `HiLDA`

`getLL` *Get the values of loglikelihood*

Description

Get the values of loglikelihood

Usage

```
getLL(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

likelihood values estimated by `pmgetSignature` in `HiLDA`

`getLogLikelihoodC` *Calculate the value of the log-likelihood for given parameters*

Description

Calculate the value of the log-likelihood for given parameters

Usage

```
getLogLikelihoodC(
  vPatternList,
  vSparseCount,
  vF,
  vQ,
  fdim,
  signatureNum,
  sampleNum,
  patternNum,
  samplePatternNum,
  isBackground,
  vF0
)
```

Arguments

`vPatternList`	The list of possible mutation features (converted to a vector)
`vSparseCount`	The table showing (mutation feature, sample, the number of mutation) (converted to a vector)
`vF`	F (converted to a vector)
`vQ`	Q (converted to a vector)

`fdim`	a vector specifying the number of possible values for each mutation signature
`signatureNum`	the number of mutation signatures
`sampleNum`	the number of cancer genomes
`patternNum`	the number of possible combinations of all the mutation features
`samplePatternNum`	the number of possible combination of samples and mutation patternns
`isBackground`	the logical value showing whether a background mutaiton features is included or not
`vF0`	a background mutaiton features

Value

a value

`getSamplelist` *Get the sample list*

Description

Get the sample list

Usage

```
getSamplelist(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

the sample list of named elements.

`getSamplelistG` *Get the sample list*

Description

Get the sample list

Usage

```
getSamplelistG(object)
```

Arguments

`object`	the MutationFeatureData class

Value

the sample list of named elements.

`getSignatures` *Get an array of signature feature distributions*

Description

Get an array of signature feature distributions

Usage

```
getSignatures(object)
```

Arguments

`object`	the EstimatedParameters class (the result of pmgetSignature)

Value

an array of signature feature distributions

`getTranscription` *Get the statsus of specifying the transcription bias*

Description

Get the statsus of specifying the transcription bias

Usage

```
getTranscription(object)
```

Arguments

`object`	the MutationFeatureData class

Value

the status of specifying the transcription bias

`select_kth_fold` *Output the training data or test data*

Description

Output the training data or test data

Usage

```
select_kth_fold(inputG, k, f_s, folds, include)
```

Arguments

`inputG`	a MutationFeatureData S4 class output by the pmsignature.
`k`	an integer number of the number of cross-validation folds.
`f_s`	a primary key of combining the feature pattern and sample ID.
`folds`	the assignment to each fold.
`include`	a boolean indictor of whether to include kth fold or not.

Value

a MutationFeatureData S4 class of either include or exclude kth fold.

`splitG` *Output the maximum potential scale reduction statistic of all parameters estimated*

Description

Output the maximum potential scale reduction statistic of all parameters estimated

Usage

```
splitG(inputG, Kfold = 3)
```

Arguments

`inputG`	a MutationFeatureData S4 class output by the pmsignature.
`Kfold`	an integer number of the number of cross-validation folds.

Value

a matrix made of perplexity from the results of cross-validation.

Examples

```
load(system.file("extdata/sample.rdata", package = "selectKSigs"))
G_split <- splitG(G, Kfold = 3)
```

Index