Package 'FLCNA'

April 29, 2022

Type Package

Title
Simultaneous CNV Detection And Subclone Clustering With Single Cell Sequencing Data

Version 1.0

Date 2022-04-19

Authors Fei Qin, Guoshuai Cai, Feifei Xiao

Description
We developed, FLCNA, a CNA detection method based on fused lasso model, which can simultaneously identify subclones in scDNA-seq data.

License GPL-3

LazyData TRUE

Depends R (¿= 4.0),
mvtnorm,
stats,
Rcpp

URL https://github.com/FeiQin92/FLCNA

RoxygenNote 7.1.2

VignetteBuilder knitr

Encoding UTF-8

Language en-GB

Suggests rmarkdown

R topics documented:

`CNA.out` *Shared CNA output*

Description

This function clusters the identified change-points to make final CNA calling. The potential CNA segments between two neighbor candidate change-points are assigned to different copy number states according to the estimated mean matrix from FLCNA R function. We use three clusters including duplication, normal state and deletion. A Gaussisan Mixture Model based clustering strategy was applied to assign each segment to the most likely cluster/state.

Usage

```
CNA.out(mean.matrix, ref, cutoff = 0.35, L = 100)
```

Arguments

`mean.matrix`	The cluster mean matrix estimated from FLCNA R function.
`cutoff`	Cutoff value to further control the number of CNAs, the larger value of cutoff, the smaller number of CNAs. The default is 0.35.
`L`	Repeat times in the EM algorithm, defaults to 100.

Value

The return is the clustered CNA segments with start position, end position and copy number states.

`state`	The CNA states assigned.
`start`	The start point of CNAs.
`end`	The end point of CNAs.
`chr`	Chromosome of CNAs.
`width`	The width of CNAs.

CNA.out.eachcell *CNA output for each cell*

Description

This function clusters the identified change-points to make final CNA calling for each cell. The potential CNA segments between two neighbor candidate change-points are assigned to different copy number states according to the estimated mean matrix from FLCNA R function and log2R data for each cell. We use three clusters including duplication, normal state and deletion. A Gaussisan Mixture Model based clustering strategy was applied to assign each segment to the most likely cluster/state.

Usage

```
CNA.out.eachcell(mean.matrix, log2R.NRC, cluster.index, cutoff = 0.5, L = 100)
```

Arguments

mean.matrix	The cluster mean matrix estimated from FLCNA R function.
log2R.NRC	Log2R data from normalization of original read counts.
cluster.index	Cluster index for all the cells.
cutoff	Cutoff value to further control the number of CNAs, the larger value of cutoff, the smaller number of CNAs. The default is 0.35.
L	Repeat times in the EM algorithm, defaults to 100.

Value

The return is the clustered CNA segments by presenting the start position and end position using CNA marker index, and the copy number states.

state	The CNA states assigned.
start	The start point for CNAs.
end	The end point for CNAs.
width	The width for CNAs.
sample	Sample index.

CNAcluster *CNAcluster*

Description

This function clusters CNAs into different states using a Gaussian Mixed Model based clustering strategy.

Usage

```
CNAcluster(Y, cp, L)
```

Arguments

Y	The numeric vector of the intensities of markers, which is the estimated mean vector in our study.
cp	The numeric vector of the position index for the identified change-points.
L	Repeat times in the EM algorithm, defaults to 100.

Value

The return is the clustered CNA segments with the start position and end position, length of the CNA and the copy number states (duplication or deletion). It also returns a vector of final candidates of change-points.

newcp	The final list of change-points.
h	The bandwidth used for the identification of change-points.
CNA.state	Copy number state for each CNA.
CNA.start	Start position of each CNA.
CNA.end	End position of each CNA.

CorrectGC *GC content correction*

Description

Normalization according to GC content.

Usage

```
CorrectGC(RC, GCContent, step)
```

Arguments

RC	A p-dimensional data vector. The read counts for a sample.
GCContent	A p-dimensional vector with gc concent.
step	Step value, a constant, used in the GC correlation procedure.

Value

The output for GC content correlation.

`CorrectMAP` *Mapp correction*

Description

Normalization according to mappability.

Usage

```
CorrectMAP(RC, MAPContent, step)
```

Arguments

`RC`	A P-dimensional data vector. The read counts for a sample.
`MAPContent`	A p-dimensional vector with mappability.
`step`	Step value, a constant, used in the Mapp correlation procedure.

Value

The output Mapp correlation.

`CorrectSize` *bin size correction*

Description

Normalization according to bin size. Since bin size is consistant in all the markers, bin size will not affect normalization results in this study.

Usage

```
CorrectSize(RC, L, step)
```

Arguments

`RC`	A P-dimensional data vector. The read counts for a sample.
`L`	The bin size used in the data the default is 100,000.
`step`	Step value, a constant, used in the bin size correlation procedure.

Value

The output bin size correlation.

`count.mu` *count.mu*

Description

Computing the number of unique cluster means for each dimension, which was used in computing BIC or GIC.

Usage

```
count.mu(mu.j, eps.diff)
```

Arguments

`mu.j`	Mean vector.
`eps.diff`	Lower bound of mean difference.

`dmvnorm_log` *dmvnorm_log*

Description

Used in sapply to find all the densities

Usage

```
dmvnorm_log(index, mu, sigma, y)
```

Arguments

`index`	Row index of mu.
`mu`	K by p matrix, each row represents one cluster mean.
`sigma`	p by p covariance matrix (assume same covariance for each cluster).
`y`	n by p data matrix.

FLCNA *FLCNA analysis*

Description

Simultaneous CNA detection and subclone identification using single cell DNA sequencing data.

Usage

```
FLCNA(
  tuning = NULL,
  K = NULL,
  lambda = c(5),
  Y,
  N = 100,
  kms.iter = 100,
  kms.nstart = 100,
  adapt.kms = FALSE,
  eps.diff = 1e-05,
  eps.em = 1e-05,
  iter.LQA = 20,
  eps.LQA = 1e-05,
  cutoff = 0.5,
  L = 100,
  model.crit = "bic"
)
```

Arguments

tuning	A 2-dimensional vector or a matrix with 2 columns, the first column is the number of clusters K and the second column is the tuning parameter λ in the penalty term. If this is missing, then K and lambda must be provided.
K	The number of clusters K.
lambda	The tuning parameter λ in the penalty term. The default is 5.
Y	A p-dimensional data matrix. Each row is an observation.
N	The maximum number of iterations in the EM algorithm. The default value is 100.
kms.iter	The maximum number of iterations in kmeans algorithm for generating the starting value for the EM algorithm.
kms.nstart	The number of starting values in K-means.
adapt.kms	A indicator of using the cluster means estimated by K-means to calculate the adaptive parameters. The default value is FALSE.
eps.diff	The lower bound of pairwise difference of two mean values. Any value lower than it is treated as 0.
eps.em	The lower bound for the stopping criterion in the EM algorithm.
iter.LQA	The number of iterations in the estimation of cluster means by using the local quadratic approximation (LQA).

`eps.LQA`	The lower bound for the stopping criterion in the estimation of cluster means.
`cutoff`	Cutoff value to further control the number of CNAs besed on mean matrix from FL model. Larger cutoff value, less CNAs.
`L`	Repeat times in the EM algorithm while outputing CNA data, defaults to 100.
`model.crit`	The criterion used to select the number of clusters K. It is either 'bic' for Bayesian Information Criterion or 'gic' for Generalized Information Criterion.

Value

This function returns the esimated parameters and some statistics of the optimal model within the given K and λ, which is selected by BIC when `model.crit = 'bic'` or GIC when `model.crit = 'gic'`.

`K.best`	The optimal number of clusters.
`mu.hat.best`	The estimated cluster means in the optimal model.
`sigma.hat.best`	The estimated covariance in the optimal model.
`alpha.hat.best`	posterior probabilities in the optimal model.
`p.hat.best`	The estimated cluster proportions in the optimal model.
`s.hat.best`	The clustering assignments using the optimal model.
`lambda.best`	The value of tuning hyperparameter lambda that provide the optimal model.
`gic.best`	The GIC of the optimal model.
`bic.best`	The BIC of the optimal model.
`llh.best`	The log-likelihood of the optimal model.
`ct.mu.best`	The degrees of freedom in the cluster means of the optimal model.
`K`	The input k values.
`lambda`	The input lambda values.
`mu.hat`	The estimated cluster means for each parameter combination.
`sigma.hat`	The estimated covariance for each parameter combination.
`p.hat`	The estimated cluster proportions for each parameter combination.
`s.hat = s.hat`	The clustering assignments for each parameter combination.
`gic`	The GIC values for each parameter combination.
`bic`	The BIC values for each parameter combination.
`llh`	The log-likelihood values for each parameter combination.
`ct.mu`	The degrees of freedom in the cluster means for each parameter combination.

Examples

```
Y <- matrix(rnorm(10000, 0, 0.5),10, 1000)
output <- FLCNA(K = c(1:2), lambda = c(2,3), Y=Y)
output
```

FLCNA_LQA *Esimation of mean matrix*

Description

Esimation of mean matrix based on local quadratic approximation (LQA).

Usage

```
FLCNA_LQA(
  k,
  K1,
  index.max,
  y,
  mu.t.all,
  mu.no.penal,
  sigma.all,
  alpha,
  lambda,
  iter.num = 20,
  eps.LQA = 1e-05,
  eps.diff = 1e-05
)
```

Arguments

k	k-th cluster index.
K1	K1 cluster in total.
index.max	Initial cluster index assigned accroding to posterior probability.
y	n by p data matrix.
mu.t.all	K by p mean matrix from previous EM-step.
mu.no.penal	K by p mean matrix of unpenalized estimates (lambda=0).
sigma.all	p by p diagnal covariance matrix.
alpha	n by K posterior probability matrix.
lambda	Tuning parameter.
iter.num	Max iterations in local quadratic approximation (LQA).
eps.LQA	LQA stop criterion.
eps.diff	Lower bound of mean difference.

Value

Estimated mu hat for k-th cluster with totally K1 cluster.

FLCNA_normalization *FLCNA normalization*

Description

Normalization function used in FLCNA.

Usage

```
FLCNA_normalization(Y, bin_size = 1e+05, gc, map)
```

Arguments

Y	A p-dimensional data matrix. Each row is an observation.
bin_size	The bin size used in the data the default is 100,000.
gc	A p-dimensional vector with gc concent.
map	A p-dimensional vector with mappability.

Value

The log2Rdata used for main step for the FLCNA method.

FLCNA_QC *FLCNA_QC*

Description

Perform QC step on single cells and bins.

Usage

```
FLCNA_QC(Y_raw, ref_raw, mapp_thresh = 0.9, gc_thresh = c(20, 80))
```

Arguments

Y_raw	raw read count matrix.
ref_raw	raw GRanges object with corresponding GC content and mappability for quality control.
mapp_thresh	scalar variable specifying mappability of each genomic bin. Default is 0.9.
gc_thresh	vector specifying the lower and upper bound of GC content threshold for quality control. Default is 20-80.

Value

A list with components after quality control.

Y	Read depth matrix after quality control.
ref	A GRanges object specifying whole genomic bin positions after quality control.

gausianMixture *Gaussian Mixture Model for CNA clustering*

Description

Gaussian Mixture Model is applied to assign each segment to the most likely cluster/state.

Usage

```
gausianMixture(x, cp, priors, L, st)
```

Arguments

x	The vector of the estimated mean of markers.
cp	The vector of the marker index of the identified change-points.
priors	Given initial parameters for the EM algorithm.
L	Repeat times in the EM algorithm. Defaults to 100.
st	Number of assumed states in the EM algorithm.

Value

The return is the clustered CNA segments with the start position and end position using CNA marker index, and the copy number states. It also returns a vector of final candidates of change-points.

p.final	Probability of falling into each state for each CNA segment after convergence.
mu.final	Segment means of each state after convergence.
cp.final	List of change-points after EM algorithm.
index.final	The index of change-points.
state.new	Assigned copy number state for each CNA.

getState *CNA states*

Description

This function uses output of Gaussian Mixture Model to obtain different CNA states.

Usage

```
getState(EM = EM)
```

Arguments

EM	The output of Gaussian Mixture Model for clustering.

Value

The return is the estimated CNA information.

`CNA.state`	Copy number state for each CNA.
`CNA.start`	Start position of each CNA.
`CNA.end`	End position of each CNA.

`nopenalty` *Clustering without penalty term*

Description

This function estimates parameters under the framework of classical mixture models without penalty term.

Usage

```
nopenalty(
  K,
  y,
  N = 100,
  kms.iter = 100,
  kms.nstart = 100,
  eps.diff = 1e-05,
  eps.em = 1e-05,
  model.crit = "gic"
)
```

Arguments

`K`	A vector of the number of clusters.
`y`	A p-dimensional data matrix. Each row is an observation.
`N`	The maximum number of iterations in the EM algorithm. The default value is 100.
`kms.iter`	The maximum number of iterations in the K-means algorithm whose outputs are the starting values for the EM algorithm.
`kms.nstart`	The number of starting values in K-means.
`eps.diff`	The lower bound of pairwise difference of two mean values. Any value lower than it is treated as 0.
`eps.em`	The lower bound for the stopping criterion.
`model.crit`	The criterion used to select the number of clusters K. It is either 'bic' for Bayesian Information Criterion or 'gic' for Generalized Information Criterion.

Details

This function estimates parameters μ, Σ, π and the clustering assignments in the model with penalty term,

$$y \sim \sum_{k=1}^{K} \pi_k f(y|\mu_k, \Sigma)$$

where $f(y|\mu_k, \Sigma_k)$ is the density function of Normal distribution with mean μ_k and variance Σ. Here we assume that each cluster has the same diagonal variance.

Value

This function returns the esimated parameters and some statistics of the optimal model within the given K and λ, which is selected by BIC when `model.crit = 'bic'` or GIC when `model.crit = 'gic'`.

`mu.hat.best`	The estimated cluster means.
`sigma.hat.best`	The estimated covariance.
`p.hat.best`	The estimated cluster proportions.
`s.hat.best`	The clustering assignments.
`K.best`	The value of K that provides the optimal model.
`llh.best`	The log-likelihood of the optimal model.
`gic.best`	The GIC of the optimal model.
`bic.best`	The BIC of the optimal model.
`ct.mu.best`	The degrees of freedom in the cluster means of the optimal model.

References

Fraley, C., & Raftery, A. E. (2002) Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association* **97(458)**, 611–631.

`updateEM` *Update parameters using EM algorithm*

Description

In the Gaussian Mixture Model, parameters will be updated based on EM algorithm.

Usage

```
updateEM(p.in, mu.in, sigma.in, means, sum.x.sq, N, len, st)
```

Arguments

`p.in`	Initial probability for each CNA cluster.
`mu.in`	Initial mean value for each CNA cluster.
`sigma.in`	Initial variance for each CNA cluster.
`means`	Mean value vector for each segment.
`sum.x.sq`	Sum of squared mean values for each segment.
`N`	Number of candiate CNAs.
`len`	Width of candiate CNAs.
`st`	Number of assumed states in the EM algorithm.

Value

The return is the updated parameters using EM algorithm

`p.new`	Updated probability for each CNA cluster.
`mu.new`	Updated mean value for each CNA cluster.
`sigma.new`	Updated variance for each CNA cluster.

Index