Package 'AMAP.Seq'

February 19, 2015

Type Package

Title Compare Gene Expressions from 2-Treatment RNA-Seq Experiments

Version 1.0

Date 2012-06-13

Author Yaqing Si

Maintainer Yaqing Si <siyaqing@gmail.com>

Description An Approximated Most Average Powerful Test with Optimal FDR Control with Application to RNA-seq Data

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-06-19 14:55:47

NeedsCompilation no

R topics documented:

`AMAP.Seq.Internal` *Internal functions/data for AMAP.Seq package*

Description

Internal functions for AMAP.Seq package

MGN.EM — *Estimate the mixture gamma-normal (MGN) distribution using expectation-maximization (EM) algorithm*

Description

the MGN distribution model the joint distribution, pi(lambda,delta), by a K-component MGN distribution, and allows degenerate normal for delta when the null hypothesis is simple.

Usage

```
MGN.EM(data, nK, p0 = NULL, d0 = 0, nK0 = 0, iter.max = 10, print.steps = FALSE, MGN0 = NULL, model = NU
```

Arguments

`data` the RNA-seq data, should be the output from RNASeq.Data

`nK` the number of components in MGN. When testing for fold-changes (FC), nK includes all components, when testing for differential expression (DE), nK only includes the components that are NOT degenerated.

`p0` the proportion of null genes when testing for DE genes.

`d0` the point where 'delta' is degerated, default is 0 when testing for DE genes.

`nK0` the number of components that are degenerated when testing for DE genes.

`iter.max` maximium number of interations in the EM algorithm

`print.steps` print the esimates of MGN in each iteration step, if TRUE. Default is FALSE

`MGN0` The initialization of the MGN. It should be a data.frame with 5 columns: pr, alpha, beta, mu, sigma. The methods of moment estimation will be used if not provided.

`model` data model, can be 'nbinom' or 'poisson'. the default will be the same as in 'data'

`nMC` the number of random samples from Gamma and Normal distrubitons in the Monte-Carlo simulation.

Value

`MGN` The estimated MGN distribution, as a data.frame with 5 columns: pr, alpha,beta,mu,sigma. pr: the proportion (weight) of each component \ alpha: alpha in the Gamma distribution \ beta: beta in the Gamma distribution \ mu: mu (mean) of the Normal distribution\ sigma: sigma (standard deviation) of the Normal distribution. sigma=0 is allowed for degenerated Normal

`lam` the shrinked estimation of lambda (mean expression for each gene)

`del` the shrinked estimation of delta (log-fold change) for each gene

Examples

```
#### see examples by typing 'help(test.AMAP)'
```

`RNASeq.Data` *Standardize the data from RNA-seq experiment*

Description

Collect all necessary input data and standardize them for follow-up analysis

Usage

```
RNASeq.Data(counts, size = NULL, group, model = "nbinom", dispersion = NULL)
```

Arguments

`counts` the counts of reads mapped to the gene. input as a G X S matrix, where G is the number of genes, and S is the number of samples

`size` the normalization factors for the counts. It should be a vector with length S, for example, the total number of reads for each column. The default is Geometric Median of the counts in each column. Users can also input the 'size' as a G X S matrix, so that each cell of the 'counts' matrix has one normalization factor.

`group` a vector indicating the design of a 2-treatment assignment, for example group=c(1,1,2,2).

`model` specify the discrete probability that model the counts. We allow 'nbinom' and 'poisson' in our test, where 'nbinom' is the default choice that use negative-binomal model.

`dispersion` the dispersion parameter for each gene (each row of the counts). users can specify the estimates by their own method, or by default, we will use quasi-likelihood method to estimate a dispersion for each gene

Value

`counts` counts of reads

`size` Normalization factor of each count

`group` treatment group

`model` distribution

`dispersion` estimated dispersion parameter in the NB model. If model="poisson", dispersion=1e-4 for all genes

Examples

```
### see examples by typing 'help(test.AMAP)'
```

test.AMAP *Calculate the test statistics of the AMAP tests*

Description

Calculate the test statistics of the AMAP tests

Usage

```
test.AMAP(data, MGN, del.lim = NULL, FC = NULL, print.steps = FALSE, Integration = "MC",nMC=NULL)
```

Arguments

`data` RNA-seq data standardized by function RNASeq.Data()

`MGN` The joint distribution, pi(lambda,delta), in form of Mixture Gamam-Normal

`del.lim` An interval, for example del.lim=c(-1,1), that is the null space for delta

`FC` A number >=1 so that the test detects genes with fold-changes greater than FC. If to detect DE genes, FC=1.

`print.steps` Print the process when calculating the test statistics

`Integration` Value can be "grid" or "MC". If Integration="grid", then the integration is done by dividing the 2-D space into grids. If Integration="MC", then the integration is done by Monte Carlo sampling.

`nMC` number of data points randomly drawn from MGN distribution by Monte Carlo simulation,the default is 50000

Value

`stat` test statistics of the AMAP tests, in logarithm scale

`prob` posterior probability of the null hypothesis, equal to exp(stat)

`fdr` estiamted FDR level if the cut-off is chosen at the gene

References

Yaqing Si and Peng Liu (2012), An Optimal Test with Maximum Average Power While Controlling FDR with Application to RNA-seq Data

Examples

```
######## Please read the help instribution above and the manuscript to
######## CHOOSE PROPER PARAMETERS LIKE nK, iter.max, nK0, FC and nMC for best use of the function
set.seed(100)
data("SimuHapMap")  # a matrix 'counts' storing simulated data with 10000 genes, two treatments, of which each has
head(cbind(counts,del.true))
counts=counts[1:200,] ### use data for only 200 genes to save time for testing example
                     ### the computation usually requires tens of minutes for 10000 genes
```

```
group=rep(1:2,each=5)

### standardize the RNA-seq data

size=Norm.GMedian(counts)   ## normalizing factor using Geometric Median
mydata=RNASeq.Data(counts=counts,size=size,group=group,model=”nbinom”)

### test DE genes

decom.est=MGN.EM(mydata,nK=3,iter.max=3,nK0=3,nMC=100)
s1=test.AMAP(mydata,MGN=decom.est$MGN,FC=1.0,nMC=100)
head(s1)

### test for FC>1.1

decom.est=MGN.EM(mydata,nK=3,iter.max=3,nK0=0,nMC=100)
s2=test.AMAP(mydata,MGN=decom.est$MGN,FC=1.1,nMC=100)
head(s2)
```

Index