

# Package ‘EBSeqHMM’

April 10, 2023

**Type** Package

**Title** Bayesian analysis for identifying gene or isoform expression changes in ordered RNA-seq experiments

**Version** 1.32.0

**Date** 2015-03-21

**Author** Ning Leng, Christina Kendzierski

**Depends** EBSeq

**Maintainer** Ning Leng <lengning1@gmail.com>

## Description

The EBSeqHMM package implements an auto-regressive hidden Markov model for statistical analysis in ordered RNA-seq experiments (e.g. time course or spatial course data). The EBSeqHMM package provides functions to identify genes and isoforms that have non-constant expression profile over the time points/positions, and cluster them into expression paths.

**License** Artistic-2.0

**Collate** 'EBTest\_ext.R' 'EBHMMNBfunForMulti.R' 'EBHMMNBfun.R'  
'EBHMMNBMultiEM\_2chain.R' 'f0.R' 'LikefunNBHMM.R' 'beta.mom.R'  
'EBSeqHMMTest.R' 'GetConfidentCalls.R' 'GetDECalls.R'  
'GetAllPaths.R' 'PlotExp.R'

**BuildVignettes** yes

**biocViews** ImmunoOncology, StatisticalMethod, DifferentialExpression,  
MultipleComparison, RNASeq, Sequencing, GeneExpression,  
Bayesian, HiddenMarkovModel, TimeCourse

**git\_url** <https://git.bioconductor.org/packages/EBSeqHMM>

**git\_branch** RELEASE\_3\_16

**git\_last\_commit** 92e51ae

**git\_last\_commit\_date** 2022-11-01

**Date/Publication** 2023-04-10

**R topics documented:**

EBSeqHMM-package . . . . .	2
beta.mom . . . . .	3
EBHMMNBfun . . . . .	4
EBHMMNBfunForMulti . . . . .	5
EBHMMNBMultiEM_2chain . . . . .	7
EBSeqHMMTest . . . . .	9
EBTest_ext . . . . .	11
f0 . . . . .	12
GeneExampleData . . . . .	13
GetAllPaths . . . . .	14
GetConfidentCalls . . . . .	15
GetDECalls . . . . .	16
IsoExampleList . . . . .	17
LikefunNBHMM . . . . .	17
PlotExp . . . . .	18

<b>Index</b>	<b>20</b>
--------------	-----------

---

EBSeqHMM-package	<i>EBSeqHMM: A Bayesian approach for identifying gene-expression changes in ordered RNA-seq experiments</i>
------------------	---

---

**Description**

The EBSeqHMM package implements an auto-regressive hidden Markov model for statistical analysis in ordered RNA-seq experiments (e.g. time course or spatial course data). The EBSeqHMM package provides functions to identify genes and isoforms that have non-constant expression profile over the time points/positions, and cluster them into expression paths.

**Details**

Package:	EBSeqHMM
Type:	Package
Version:	0.99.1
Date:	2014-09-16
License:	Artistic-2.0

**Author(s)**

Ning Leng, Christina Kendziorski Maintainer: Ning Leng <nleng@wisc.edu>

## References

Leng, N., Li, Y., Mcintosh, B. E., Nguyen, B. K., Duffin, B., Tian, S., Thomson, J. A., Colin, D., Stewart, R. M., and Kendziorski, C. (2014). Ebseq-hmm: A bayesian approach for identifying gene-expression changes in ordered rna-seq experiments.

## See Also

EBSeq

## Examples

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
EBSeqHMMGeneOut <- EBSeqHMMTest(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
                                UpdateRd=2)
```

---

beta.mom

*Method of moments estimation ( beta distribution )*

---

## Description

Method of moments estimation ( beta distribution )

## Usage

```
beta.mom(qs.in)
```

## Arguments

qs.in            A vector contains the numbers that will be fitted with a beta distribution.

## Details

beta.mom() function can be used to estimate parameters in a Beta function using method of moments

## Value

alpha.hat,beta.hat: Returns the estimation of alpha and beta.

## Author(s)

Ning Leng

## Examples

```
beta.mom(rbeta(10,1,1))
```

EBHMMNBfun

*Baum-Welch algorithm for a single hidden markov chain***Description**

Baum-Welch algorithm for a single hidden markov chain

**Usage**

```
EBHMMNBfun(Data,NgVector=NULL,Conditions, sizeFactors,
PriorFC=1.5,homo=TRUE, maxround=5,
Pi0=NULL, Tran=NULL,NoTrend=FALSE, NumTranStage=3,
FCParam=NULL, AlphaIn=NULL,BetaIn=NULL,
StateNames=c("Up","NC","Down"),
EM=TRUE, UpdateParam=TRUE, Print=TRUE,
OnlyQ=FALSE,WithinCondR=TRUE,
PenalizeLowMed=TRUE, PenalizeLowMedQt=.2,PenalizeLowMedVal=10)
```

**Arguments**

Data	input data, rows are genes/isoforms and columns are samples
NgVector	Ng vector; NULL for gene level data
Conditions	A factor indicates the condition (time/spatial point) which each sample belongs to.
sizeFactors	a vector indicates library size factors
Tran	initial values for transition matrices
Pi0	initial values for starting probabilities
NumTranStage	number of states
PriorFC	target FC for gradient change
StateNames	name of the hidden states
homo	whether the chain is assumed to be homogenous
maxround	max number of iteration
AlphaIn,BetaIn	If the parameters are known and the user doesn't want to estimate them from the data, user may specify them here.
NoTrend	if NoTrend=TRUE, initial transition probabilities will be set to be equal
FCParam	not in use
EM	Whether estimate the prior parameters of the beta distribution by EM
UpdateParam	Whether update starting probabilities and transition probabilities
OnlyQ	If OnlyQ=TRUE, the function will only return estimated q's.
WithinCondR	By defining WithinCondR=TRUE, estimation of r's are obtained within each condition. (WithinCondR=FALSE is not suggested here)
Print	Whether print the elapsed-time while running the test.
PenalizeLowMed, PenalizeLowMedQt, PenalizeLowMedVal	Transcripts with median quantile <= PenalizeLowMedQt will be penalized

**Details**

EBHMMNBfun() function implements the Baum-Welch algorithm that estimates the starting probabilities and transition probabilities of a single hidden Markov model. Here the emission distribution of each gene is assumed to be a Beta-Negative Binomial distribution with parameters ( $r_g$ ,  $\alpha$ ,  $\beta$ ), in which  $\alpha$  and  $\beta$  are shared by all the genes and  $r_g$  is gene specific. If not specified,  $r_g$ ,  $\alpha$  and  $\beta$  will be estimated using method of moments. For isoform data, we assume isoforms from the same Ig group share the same  $\beta^{Ig}$ .  $\alpha$  is shared by all the isoforms and  $r_{gi}$  is isoform specific. The user also needs to specify an expected FC.

**Value**

MAPTerm: the most likely path of each gene/isoform. MAPTermNum: numeric version of MAPTerm.

AllTerm: all possible expression paths considered in the model. PP: posterior probability of being each expression path.

WhichMax: index of the most likely path. Allf: prior probability of being each path.

Pi0Track: estimated starting probabilities of each iteration.

TranTrack: estimated transition probabilities of each iteration.

AlphaTrack, BetaTrack: estimated  $\alpha$  and  $\beta(s)$ .

LLAll=PostSumForLL.Sum: log likelihood of the model.

**Author(s)**

Ning Leng

**Examples**

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
tmp <- EBHMMNBfun(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
  maxround=2, OnlyQ=TRUE)
```

---

EBHMMNBfunForMulti      *Baum-Welch algorithm for multiple hidden markov chains*

---

**Description**

Baum-Welch algorithm for multiple hidden markov chains

**Usage**

```
EBHMMNBfunForMulti(Data,PPIn,
NgVector=NULL,Conditions, sizeFactors,
PriorFC=1.5,homo=TRUE, maxround=5,
Pi0=NULL, Tran=NULL, NumTranStage=3,
FCParam=NULL, AlphaIn=NULL,BetaIn=NULL,
StateNames=c("Up","NC","Down"),
EM=TRUE, UpdateParam=TRUE, Print=TRUE,WithinCondR=TRUE,
PenalizeLowMed=TRUE, PenalizeLowMedQt=.2,PenalizeLowMedVal=10)
```

**Arguments**

Data	input data, rows are genes/isoforms and columns are samples
PPIn	PPDE for all adjacent comparisons
NgVector	Ng vector; NULL for gene level data
Conditions	A factor indicates the condition (time/spatial point) which each sample belongs to.
sizeFactors	a vector indicates library size factors
Tran	initial values for transition matrices
Pi0	initial values for starting probabilities
NumTranStage	number of states in two chains
PriorFC	target FC for gradient change
StateNames	name of the hidden states
homo	whether the chain is assumed to be homogenous
maxround	max number of iteration
AlphaIn,BetaIn	If the parameters are known and the user doesn't want to estimate them from the data, user may specify them here.
FCParam	not in use
EM	Whether estimate the prior parameters of the beta distribution by EM
UpdateParam	Whether update starting probabilities and transition probabilities
WithinCondR	By defining WithinCondR=TRUE, estimation of r's are obtained within each condition. (WithinCondR=FALSE is not suggested here)
Print	Whether print the elapsed-time while running the test.
PenalizeLowMed, PenalizeLowMedQt, PenalizeLowMedVal	Transcripts with median quantile $\leq$ PenalizeLowMedQt will be penalized

**Details**

EBHMMNBfunForMulti() function implements the Balm-Welch algorithm that estimates the starting probabilities and transition probabilities of a hidden Markov model with multiple chains. Here the emission distribution of each gene is assumed to be a Beta-Negative Binomial distribution with parameters  $(r_g, \alpha, \beta)$ , in which  $\alpha$  and  $\beta$  are shared by all the genes and  $r_g$  is gene specific. If not specified,  $r_g$ ,  $\alpha$  and  $\beta$  will be estimated using method of moments. For isoform data, we assume isoforms from the same Ig group share the same  $\beta^{Ig}$ .  $\alpha$  is shared by all the isoforms and  $r_{gi}$  is isoform specific. The user also needs to specify an expected FC.

**Value**

MAPTerm: the most likely path of each gene/isoform.  
 MAPTermNum: numeric version of MAPTerm.  
 AllTerm: all possible expression paths considered in the model.  
 PP: posterior probability of being each expression path.  
 WhichMax: index of the most likely path.  
 Allf: prior probability of being each path.  
 Pi0Track: estimated starting probabilities of each iteration.  
 TranTrack: estimated transition probabilities of each iteration.  
 AlphaTrack, BetaTrack: estimated alpha and beta(s).  
 LLAll=PostSumForLL.Sum: log likelihood of the model.

**Author(s)**

Ning Leng

**Examples**

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
tmp <- EBHMMNBfunForMulti(Data=GeneExampleData, PPIn=matrix(1,ncol=15, nrow=100),sizeFactors=Sizes, Conditions=Conditions,
  maxround=2)
```

---

EBHMMNBMultiEM\_2chain *Run EBSeqHMM model with a fixed expected FC*

---

**Description**

Run EBSeqHMM model with a fixed expected FC

**Usage**

```
EBHMMNBMultiEM_2chain(Data,
  NgVector=NULL, Conditions, AllTran=NULL,
  AllPi0=NULL, Terms=NULL,
  sizeFactors, NumTranStage=c(3,2),PriorFC=2,
  StateNames=c("Up","Down"),homo=FALSE,
  UpdateRd=5, PIBound=.9, UpdatePI=FALSE,Print=FALSE,
  WithinCondR=TRUE,
  PenalizeLowMed=TRUE, PenalizeLowMedQt=.1,PenalizeLowMedVal=10)
```

**Arguments**

Data	input data, rows are genes and columns are samples
NgVector	Ng vector; NULL for gene level data
Conditions	A factor indicates the condition (time/spatial point) which each sample belongs to.
AllTran	initial values for transition matrices
AllPi0	initial values for starting probabilities
Terms	Terms
sizeFactors	a vector indicates library size factors
StateNames	names of the hidden states
NumTranStage	number of states in two chains
PriorFC	target FC for gradient change
homo	whether the chain is assumed to be homogenous
UpdateRd	max number of iteration
UpdatePI	whether update the mixture proportion of two chains
PIBound	upper bound of the mixture proportion of the two chains
Print	Whether print the elapsed-time while running the test.
WithinCondR	By defining WithinCondR=TRUE, estimation of r's are obtained within each condition. (WithinCondR=FALSE is not suggested here)
PenalizeLowMed, PenalizeLowMedQt, PenalizeLowMedVal	Transcripts with median quantile $\leq$ PenalizeLowMedQt will be penalized

**Details**

EBHMMNBMultiEM\_2chain() function implements the EBSeqHMM model to perform statistical analysis in an RNA-seq experiment with ordered conditions. EBHMMNBMultiEM\_2chain() calls EBHMMNBfunForMulti() function to perform Balm-Welch algorithm that estimates the starting probabilities and transition probabilities. Here the emission distribution of each gene is assumed to be a Beta-Negative Binomial distribution with parameters ( $r_g$ ,  $\alpha$ ,  $\beta$ ), in which  $\alpha$  and  $\beta$  are shared by all the genes and  $r_g$  is gene specific. If not specified,  $r_g$ ,  $\alpha$  and  $\beta$  will be estimated using method of moments. For isoform data, we assume isoforms from the same Ig group share the same  $\beta^{Ig}$ .  $\alpha$  is shared by all the isoforms and  $r_{gi}$  is isoform specific. The user also needs to specify an expected FC. Function EBSeqHMMTest() runs several models with varying FCs and returns the model with maximum likelihood.

**Value**

Pi0Out: estimated starting probabilities of each iteration.  
 TranOut: estimated transition probabilities of each iteration.  
 Pi: estimated probability of being each chain.  
 Alpha, Beta: estimated alpha and beta(s).  
 LLSum: log likelihood of the model.



QList: estimated q's.  
 MgAllPP: marginal PP for all paths.  
 MgAllMAPChar: Most likely path based on MgAllPP.  
 MgAllMaxVal: Highest PP based on MgAllPP.  
 PPMatW: Posterior probabilities of being each of the chains.

### Author(s)

Ning Leng

### Examples

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
tmp <- EBHMMNBMultiEM_2chain(Data=GeneExampleData,sizeFactors=Sizes, Conditions=Conditions,
                             UpdateRd=2)
```

---

EBSeqHMMTest	<i>Identify DE genes and classify them into their most likely path in an RNA-seq experiment with ordered conditions</i>
--------------	---

---

### Description

Identify DE genes and classify them into their most likely path in an RNA-seq experiment with ordered conditions

### Usage

```
EBSeqHMMTest(Data,
              NgVector=NULL, Conditions, AllTran=NULL,
              AllPi0=NULL, Terms=NULL,
              sizeFactors, NumTranStage=c(3,2),FCV=2,
              homo=FALSE, UpdateRd=10, PIBound=.9, UpdatePI=FALSE,
              Print=FALSE,WithinCondR=TRUE,Qtrm=.75,QtrmCut=10,
              PenalizeLowMed=TRUE, PenalizeLowMedQt=.1,PenalizeLowMedVal=10)
```

### Arguments

Data	input data, rows are genes and columns are samples
NgVector	Ng vector; NULL for gene level data
Conditions	A factor indicates the condition (time/spatial point) which each sample belongs to.
AllTran	initial values for transition matrices
AllPi0	initial values for starting probabilities

Terms	Terms
FCV	candidate values for expected FC. Default is 2. If user wants to search through a list of candidate FCs, he/she may define FCV as a vector. e.g. By defining <code>FCV=seq(1.4,2,.2)</code> , the function will search over (1.4 1.6 1.8 2.0). Note that searching over a number of candidate FCs will increase the running time.
sizeFactors	a vector indicates library size factors
NumTranStage	number of states in two chains
homo	whether the chain is assumed to be homogenous
UpdateRd	max number of iteration
UpdatePI	whether update the mixture proportion of two chains
PIBound	upper bound of the mixture proportion of the two chains
Qtrm,QtrmCut	Transcripts with Qtrm th quantile $\leq$ QtrmCut will be removed before testing. The default value is Qtrm = 0.75 and QtrmCut=10. By default setting, transcripts that have >75% of the samples with expression less than 10 won't be tested.
WithinCondR	By defining WithinCondR=TRUE, estimation of r's are obtained within each condition. (WithinCondR=FALSE is not suggested here)
Print	Whether print the elapsed-time while running the test.
PenalizeLowMed, PenalizeLowMedQt, PenalizeLowMedVal	Transcripts with median quantile $\leq$ PenalizeLowMedQt will be penalized

## Details

EBSeqHMMTest() function applies EBSeqHMM model with different expected FC's and select the optimal FC that maximizes the log likelihood. EBSeqHMMTest() calls EBHMMNBMultiEM\_2chain() function which implements the EBSeqHMM model to perform statistical analysis in an RNA-seq experiment with ordered conditions based on a fixed expected FC. EBSeqHMMTest() runs EBHMMNBMultiEM\_2chain() with varying FCs (default is `seq(1.4,2,.2)`). And it will return the results of the model with optimal FC. Here the emission distribution of each gene is assumed to be a Beta-Negative Binomial distribution with parameters ( $r_g$ ,  $\alpha$ ,  $\beta$ ), in which  $\alpha$  and  $\beta$  are shared by all the genes and  $r_g$  is gene specific. If not specified,  $r_g$ ,  $\alpha$  and  $\beta$  will be estimated using method of moments. For isoform data, we assume isoforms from the same Ig group share the same  $\beta^{Ig}$ .  $\alpha$  is shared by all the isoforms and  $r_{gi}$  is isoform specific. The user also needs to specify an expected FC.

## Value

Pi0Out: estimated starting probabilities of each iteration.  
 TranOut: estimated transition probabilities of each iteration.  
 Pi: estimated probability of being each chain.  
 Alpha, Beta: estimated alpha and beta(s).  
 LLSum: log likelihood of the model.  
 QList: estimated q's.  
 MgAllPP: marginal PP for all paths.  
 MgAllMAPChar: Most likely path based on MgAllPP.

MgAllMaxVal: Highest PP based on MgAllPP.

PPMatW: Posterior probabilities of being each of the chains.

FCLikelihood: log likelihood of each FC.

### Author(s)

Ning Leng

### Examples

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
EBSeqHMMGeneOut <- EBSeqHMMTest(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
                                UpdateRd=2)
```

---

EBTest\_ext

*Extended EBTest function*

---

### Description

Extended EBTest function

### Usage

```
EBTest_ext(Data,NgVector=NULL,Conditions,
           sizeFactors, maxround, Pool=FALSE, NumBin=1000,
           ApproxVal=10^-10, Alpha=NULL, Beta=NULL,
           PInput=NULL,RInput=NULL,PoolLower=.25,
           PoolUpper=.75,OnlyCalcR=FALSE,Print=TRUE)
```

### Arguments

Data	Input data, rows are genes/isoforms and columns are samples. Data should come from a two condition experiment
NgVector	Ng vector; NULL for gene level data
Conditions	A factor indicates the condition (time/spatial point) which each sample belongs to. Only two levels are allowed.
sizeFactors	a vector indicates library size factors
maxround	number of iteration
Pool	While working without replicates, user could define the Pool = TRUE in the EBTest function to enable pooling.
NumBin	By defining NumBin = 1000, EBSeq will group the genes with similar means together into 1,000 bins.

PoolLower, PoolUpper

With the assumption that only subset of the genes are DE in the data set, we take genes whose FC are in the PoolLower - PoolUpper quantile of the FCs as the candidate genes (default is 25 bin, the bin-wise variance estimation is defined as the median of the cross condition variance estimations of the candidate genes within that bin. We use the cross condition variance estimations for the candidate genes and the bin-wise variance estimations of the host bin for the non-candidate genes.

ApproxVal      The variances of the transcripts with mean < var will be approximated as mean/(1-ApproxVal).

Alpha, Beta, PInput, RInput

If the parameters are known and the user doesn't want to estimate them from the data, user may specify them here.

Print            Whether print the elapsed-time while running the test.

OnlyCalcR      if OnlyCalcR=TRUE, the function will only return estimation of r's.

## Details

EBSeq\_ext() function is an extension of EBTest() function, which is used to calculate the conditional probability  $P(X_{g,t} | X_{g,t-1})$ . In EBSeqHMM, we assume the conditional distribution is Beta-Negative Binomial.

## Value

See [EBTest](#)

## Author(s)

Ning Leng

## Examples

```
data(GeneExampleData)
Data=GeneExampleData[,1:6]
CondVector <- rep(paste("t",1:2,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2"))
Sizes <- MedianNorm(Data[1:10,])
Out <- EBTest_ext(Data=Data[1:10,], sizeFactors=Sizes, Conditions=Conditions,
  maxround=1)
```

---

f0

*Calculate the prior predictive distribution of the Beta-Negative Binomial model*

---

## Description

Calculate the prior predictive distribution of the Beta-Negative Binomial model

**Usage**

```
f0(Input, AlphaIn, BetaIn, EmpiricalR, NumOfGroups, log)
```

**Arguments**

Input	expression values
AlphaIn, BetaIn, EmpiricalR	The parameters estimated from last iteration of EM.
NumOfGroups	How many transcripts within each Ng group
log	If set as TRUE, the output will in log scale.

**Details**

Function f0() will calculate the Beta-Negative Binomial prior predictive probability for a given set of parameters

**Value**

output a numeric vector, each element shows the prior predictive probability of one gene/isoform

**Author(s)**

Ning Leng

**Examples**

```
f0(matrix(rnorm(100,100,1),ncol=10), .5, .6,
      matrix(rnorm(100,200,1),ncol=10), 100, TRUE)
```

---

GeneExampleData

*Simulated gene level data set with 5 ordered conditions*

---

**Description**

'GeneExampleData' gives the gene level simulated data with 5 ordered conditions, triplicates for each condition. The data set was simulated following the Negative Binomial distribution. The parameters of each gene (mean and overdispersion) were sampled from the empirical estimates from an empirical RNA-Seq data set from Thomson lab at Morgridge Institute for Research.

**Format**

GeneExampleData is a matrix with 100 genes (rows) and 15 samples (columns).

**See Also**

IsoExampleList

**Examples**

```
data(GeneExampleData)
str(GeneExampleData)
```

---

GetAllPaths	<i>Obtain all possible gene paths for an RNA-seq experiments with ordered conditions</i>
-------------	--

---

**Description**

Obtain all possible gene paths for an RNA-seq experiments with ordered conditions

**Usage**

```
GetAllPaths(EBSeqHMMOut, OnlyDynamic=TRUE)
```

**Arguments**

EBSeqHMMOut	output from EBSeqHMMTest function
OnlyDynamic	if specifies as TRUE, only dynamic paths will be shown

**Details**

GetAllPaths() function may be used to generate all possible expression paths of a particular design.

**Value**

output: a vector of paths. For example, Up-Up-Up-Up, Up-Up-EE-EE, Up-Down-Up-EE, etc.

**Author(s)**

Ning Leng

**Examples**

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
EBSeqHMMGeneOut <- EBSeqHMMTest(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
                                UpdateRd=2)
AllPaths <- GetAllPaths(EBSeqHMMGeneOut)
```

---

GetConfidentCalls      *Obtain confident gene calls for classifying genes into expression paths*

---

### Description

Obtain confident gene calls for classifying genes into expression paths

### Usage

```
GetConfidentCalls(EBSeqHMMOut, FDR=.05, cutoff=0.5, OnlyDynamic=TRUE, Paths=NULL)
```

### Arguments

EBSeqHMMOut	output from EBSeqHMMTest function
FDR	Target FDR, default is 0.05.
cutoff	cutoff to use for defining a confident call. Genes with PP_path greater or equal to cutoff will be called as a confident call. Default is 0.5.
OnlyDynamic	if specifies as T, only dynamic paths will be shown
Paths	paths that are of interest. Default is NULL. If it is not specified, all possible paths will be considered.

### Details

Function GetConfidentCalls() can be used to obtain a list of DE genes/isoforms with user specific cutoffs. To obtain a list of DE genes/isoforms with a target FDR alpha, the user may specify FDR=alpha. To further choose genes/isoforms with high posterior probability of being its most likely path, the user may specify the option cutoff (default is 0.5). Then genes or isoforms with PP(most likely path) >= 0.5 will be selected

### Value

Overall: a list of genes/isoforms that are identified as DE under the target FDR, shown are their names and PPs; EachPath: a list object, each sublist contains confident calls (genes/isoforms) that have PP(path)>=cutoff for a particular expression path, shown are their names and PPs; NumEach: length of each sublist in EachPath. EachPathName: gene/isoform names in each of the sublists in EachPath

### Note

Output: output a list of genes that are classified to a expression path as a confident assignment.

### Author(s)

Ning Leng

**Examples**

```

data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
EBSeqHMMGeneOut <- EBSeqHMMTest(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
                                UpdateRd=2)
GeneDECalls <- GetDECalls(EBSeqHMMGeneOut, FDR=.05)
GeneConfCalls <- GetConfidentCalls(EBSeqHMMGeneOut, FDR=.05,cutoff=.5, OnlyDynamic=TRUE)

```

---

GetDECalls

*Obtain DE gene/isoform list at a certain FDR*


---

**Description**

Obtain DE gene/isoform list at a certain FDR

**Usage**

```
GetDECalls(EBSeqHMMOut, FDR=.05)
```

**Arguments**

EBSeqHMMOut	output from EBSeqHMMTest function
FDR	Target FDR; default is 0.05

**Details**

Function GetDECalls() can be used to obtain a list of DE genes/isoforms with user specific cutoffs. To obtain a list of DE genes/isoforms with a target FDR alpha, the user may specify FDR=alpha.

**Value**

a list of genes/isoforms that are identified as DE under the target FDR, shown are their names and PPs;

**Note**

Output: output a list of genes that are DE in at least one condition in an RNA-seq experiment with multiple ordered conditions

**Author(s)**

Ning Leng



## Examples

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
EBSeqHMMGeneOut <- EBSeqHMMTest(Data=GeneExampleData, sizeFactors=Sizes, Conditions=Conditions,
                                UpdateRd=2)
GeneDECalls <- GetDECalls(EBSeqHMMGeneOut, FDR=.05)
```

---

IsoExampleList

*Simulated isoform level data set with 5 ordered conditions*

---

## Description

'IsoExampleList' gives the isoform level simulated data with 5 ordered conditions, triplicates for each condition. The data set was simulated following the Negative Binomial distribution. The parameters of each isoform (mean and overdispersion) were sampled from the isoform level empirical estimates from an empirical RNA-Seq data set from Thomson lab at Morgridge Institute for Research.

## Format

IsoExampleList is a list with three components. IsoExampleList\$IsoExampleData contains a matrix with 200 isoform (rows) and 15 samples (columns). IsoExampleList\$IsoNames contains a vector of isoform names. IsoformExampleList\$IsosGeneNames contains a vector indicating the gene each isoform belongs to.

## See Also

GeneExampleData

## Examples

```
data(IsoExampleList)
str(IsoExampleList)
```

---

LikefunNBHMM

*Likelihood function of the Beta-Negative Binomial HMM Model*

---

## Description

Likelihood function of the Beta-Negative Binomial HMM Model

## Usage

```
LikefunNBHMM(ParamPool, InputPool)
```

**Arguments**

ParamPool        The parameters that will be estimated in EM.  
 InputPool        The control parameters that will not be estimated in EM

**Details**

The likelihood function of the Beta-Negative Binomial HMM model used in EBSeqHMM. EBSeqHMM uses `optim()` function to obtain the optimal estimates that minimizes the likelihood.

**Value**

optimal estimates of the parameters of interest

**Author(s)**

Ning Leng

**Examples**

```
data(GeneExampleData)
tmp <- GeneExampleData[1:10,]
In <- list(tmp,1,5,10,3,tmp,rep(1,15),as.factor(rep(1:5,each=3)),10,cbind(rep(.5,10),rep(1,10),rep(2,10)))
Start <- c(1,1)
LikefunNBHMM(Start,In)
```

---

 PlotExp

---

*Plot expression of a single gene*


---

**Description**

Plot expression of a single gene

**Usage**

```
PlotExp(NormalizedData, Conditions, Name)
```

**Arguments**

NormalizedData   Expression data after adjusting for library size factors  
 Conditions        sample conditions  
 Name                name of the gene/isoform of interest

**Details**

PlotExp() function will generate line plots for genes or isoforms of interest.

**Value**

PlotExp() function will generate line plots for genes or isoforms of interest.

**Author(s)**

Ning Leng

**Examples**

```
data(GeneExampleData)
CondVector <- rep(paste("t",1:5,sep=""),each=3)
Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
Sizes <- MedianNorm(GeneExampleData)
NormData <- GetNormalizedMat(GeneExampleData, Sizes)
PlotExp(NormData, Conditions, "Gene_1")
```

# Index

## \* datasets

GeneExampleData, [13](#)

IsoExampleList, [17](#)

## \* package

EBSeqHMM-package, [2](#)

beta.mom, [3](#)

EBHMMNBfun, [4](#)

EBHMMNBfunForMulti, [5](#)

EBHMMNBMultiEM\_2chain, [7](#)

EBSeqHMM (EBSeqHMM-package), [2](#)

EBSeqHMM-package, [2](#)

EBSeqHMMTest, [9](#)

EBTest, [12](#)

EBTest\_ext, [11](#)

f0, [12](#)

GeneExampleData, [13](#)

GetAllPaths, [14](#)

GetConfidentCalls, [15](#)

GetDECalls, [16](#)

IsoExampleList, [17](#)

LikefunNBHMM, [17](#)

PlotExp, [18](#)