Package 'seq2pathway'

April 16, 2019

Type Package

Title a novel tool for functional gene-set (or termed as pathway) analysis of next-generation sequencing data

Version 1.14.0

Date 2015-09-18

Author Xinan Yang <xyang2@uchicago.edu>; Bin Wang <binw@uchicago.edu>

Maintainer Xinan Yang <xyang2@uchicago.edu> with contribution from Lorenzo Pesce <lpesce@uchicago.edu> and Ana Marija Sokovic <anamarija@uchicago.edu>

Depends R (>= 2.10.0)

biocViews Software

Imports nnet, WGCNA, GSA, biomaRt, GenomicRanges, seq2pathway.data

Description Seq2pathway is a novel tool for functional gene-set (or termed as pathway) analysis of next-generation sequencing data, consisting of ``seq2gene'' and ``gene2path'' components. The seq2gene links sequence-level measurements of genomic regions (including SNPs or point mutation coordinates) to gene-level scores, and the gene2pathway summarizes gene scores to pathway-scores for each sample. The seq2gene has the feasibility to assign both coding and non-exon regions to a broader range of neighboring genes than only the nearest one, thus facilitating the study of functional non-coding regions. The gene2pathway takes into account the quantity of significance for gene members within a pathway compared those outside a pathway. The output of seq2pathway is a general structure of quantitative pathway-level scores, thus allowing one to functional interpret such datasets as RNA-seq, ChIP-seq, GWAS, and derived from other next generational sequencing experiments.

License GPL-2

NeedsCompilation no

git_url https://git.bioconductor.org/packages/seq2pathway

git_branch RELEASE_3_8

git_last_commit 11464ae

git_last_commit_date 2018-10-30

Date/Publication 2019-04-15

R topics documented:

`addDescription` *Retrieve "gene description" attributes for gene symbol.*

Description

A function wrappered from Rpackage "biomaRt". Get gene description information from gene symbol information.

Usage

```
addDescription(genome=c("hg19","mm10","mm9"), genevector)
```

Arguments

`genome`	A character specifies the genome type. Currently, choice of "hg19", "mm10", and "mm9" is supported.
`genevector`	A characteristic vector of gene symbols.

Value

A characteristic matrix of gene symbols and descriptions.

Author(s)

Bin Wang

References

Durinck S, Spellman P, Birney E and Huber W (2009) Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. *Nature Protocols*, **4**, 1184–1191.

Durinck S, Moreau Y, Kasprzyk A, Davis S, De Moor B, Brazma A and Huber W (2005) BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. *Bioinformatics*, **21**, 3439–3440.

Examples

```
require(biomaRt)
data(dat_chip)
gene_description<-addDescription(genome="hg19",genevector=rownames(dat_chip)[1:3])
```

Chipseq_Peak_demo *chip seq loci data example*

Description

chip seq loci data example

Usage

```
data("Chipseq_Peak_demo")
```

Format

A data frame with 5 observations on the following 5 variables.

`peakID` unique chip peak name information

`chrom` chromosome information

`start` loci start

`end` loci end

`signalvalue` a numeric vector

Value

a data frame of chip sequence peak information

Examples

```
data(Chipseq_Peak_demo)
head(Chipseq_Peak_demo)
```

dat_chip *chip seq data example*

Description

chip seq data example

Usage

```
data("dat_chip")
```

Format

A data frame with 639 observations on the following 1 variables.

`peakscore` a numeric vector

Value

A data frame of single sample gene scores.

Examples

```
data(dat_chip)
```

`dat_RNA` *RNA sequence data example*

Description

RNA sequence data example

Usage

```
data("dat_RNA")
```

Format

A data frame with 5000 observations on the following 5 variables.

`TCGA_2841` a numeric vector

`TCGA_2840` a numeric vector

`TCGA_2843` a numeric vector

`TCGA_2842` a numeric vector

`TCGA_2845` a numeric vector

Value

A data frame of 5 sample gene scores.

Examples

```
data(dat_RNA)
```

`FisherTest_GO_BP_MF_CC`

A wrapper function to perform the Fisher's exact test, using GO-defined genesets.

Description

A wrapper function to perform the Fisher's exact test, using GO-defined genesets.

Usage

```
FisherTest_GO_BP_MF_CC(gs, genome=c("hg38","hg19","mm10","mm9"),
                       min_Intersect_Count=5,
                       Ontology=c("GOterm", "BP","MF", "CC","newOntology"),
  newOntology=NULL)
```

Arguments

gs	A characteristic vector of gene symbols, the input gene list.
genome	A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.
min_Intersect_Count	A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.
Ontology	A character specifies the Gene Ontology, choice of "GOterm", "BP","MF", "CC" and "newOntology" is supported.
newOntology	A list of two lists with the same ontology IDs. or each ontology ID, the 1st list is the lists of defined genes and the 2nd list is the desceiption.

Value

A list of 3 data frames, each is a result of Fisher's exact test, using GO CC, BP, MF respectively. Each data frame reports FET results with the following columns.

GOID	GO term IDs
Description	GO definition and description for the gene-sets
Fisher_Pvalue	is the raw P-values
Fisher_odds	estimate of the odds ratios
FDR	the multi-test adjusted P-values using the Benjamini and Hochberg method
Intersect_Count	the sizes of overlap between GO gene members and the input genelist
GO_gene_inBackground	the counts of genes among each GO term that are also within the given genome background
GO_gene_raw_count	the original counts of genes in each GO term
Intersect_gene	the intersecting genes' symbols

Author(s)

Bin Wang, Xinan Yang

Examples

```
data(dat_chip)
head(dat_chip)
data(GO_BP_list,package="seq2pathway.data")
data(Des_BP_list,package="seq2pathway.data")
newOntology <- list(GO_BP_list[1:200], Des_BP_list[1:200])
# A demo run of this funcion
FS_test<- FisherTest_GO_BP_MF_CC(gs=as.vector(rownames(dat_chip)),
                                 Ontology="newOntology", newOntology=newOntology)
FS_test

## Not run:
data(dat_chip)
FS_test<-FisherTest_GO_BP_MF_CC(gs=rownames(dat_chip)[1:20], genome="hg19",
         min_Intersect_Count=1, Ontology="BP")
```

```
  FS_test$GO_BP[1:3,]
## End(Not run)
```

`FisherTest_MsigDB` *A wrapper function to perform conditional Fisher's exact test, using custom-defined genesets.*

Description

A wrapper function to perform conditional FET, using custom-defined genesets.

Usage

```
FisherTest_MsigDB(gsmap, gs, genome=c("hg38","hg19","mm10","mm9"),
                  min_Intersect_Count=5)
```

Arguments

`gsmap`	An R GSA.genesets object defined by the package "GSA" for functional gene-set (or termed as pathway for simplification). User can call the GSA.read.gmt function in R GSA package to load customized gene-sets with a .gmt format.
`gs`	A characteristic vector of gene symbols, the input genelist.
`genome`	A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.
`min_Intersect_Count`	A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.

Value

A data frame of Fisher's exact tested result with the following columns:

`GeneSet`	MSigDB gene-set names (IDs)
`Description`	MSigDB definition and description for gene-sets
`Fisher_Pvalue`	the raw Pvalues
`Fisher_odds`	estimate of the odds ratios
`FDR`	the multi-test adjusted Pvalues using the Benjamini and Hochberg method
`Intersect_Count`	the sizes of the overlap between the MSigDB gene-set genes and the input genelist
`MsigDB_gene_inBackground`	the counts of genes among each MSigDB gene-set that are also within genome background
`MsigDB_gene_raw_Count`	the original counts of genes in each MSigDB gene-set
`Intersect_gene`	the intersecting genes' symbols

Author(s)

Bin Wang

Examples

```
 data(dat_chip)
 data(MsigDB_C5,package="seq2pathway.data")
 #generate a demo GSA.genesets object
 demoDB <- MsigDB_C5
 x=100
 for(i in 1:3) demoDB[[i]]<-MsigDB_C5[[i]][1:x]
 FS_test<-FisherTest_MsigDB(gsmap=demoDB,
sample(unlist(demoDB$genesets),10), genome="hg19",
min_Intersect_Count=1)
 FS_test[1:3,]
 ## Not run:
 FS_test<-FisherTest_MsigDB(gsmap=MsigDB_C5,
gs=rownames(dat_chip), genome="hg19",
min_Intersect_Count=1)

## End(Not run)
```

`gene2pathway_test` *A wrapper function to perform gene2pathway test.*

Description

The function includes two part, one runs the classical Fisher's exact test, the other runs novel gene2pathway test.

Usage

```
gene2pathway_test(dat, DataBase="GOterm", FisherTest=TRUE, EmpiricalTest=FALSE,
                  method=c("FAIME", "KS-rank", "cumulative-rank"),
                  genome=c("hg38","hg19","mm10","mm9"), alpha=5,
                  logCheck=FALSE, na.rm=FALSE, B=100, min_Intersect_Count=5)
```

Arguments

`dat`	A data frame of gene expression or a matrix of sequencing-derived gene-level measurements. The rows of dat correspond to genes, and the columns correspond to sample profile (eg. Chip-seq peak scores, somatic mutation p-values, RNS-seq or micro-array gene expression values). Note that the rows must be labeled by official gene symbol. The values contained in dat should be either finite or NA.
`DataBase`	A character string assigns an R GSA.genesets object to define gene-set. User can call GSA.read.gmt to load customized gene-sets with a .gmt format. If not specified, GO defined gene sets (BP,MF,CC) will be used.
`FisherTest`	A Boolean value. By default is TRUE to excute the function of the Fisher's exact test. Otherwise, only excutes the function of gene2pathway test.
`EmpiricalTest`	A Boolean value. By default is FALSE for multiple-sample dat. When true, gene2pathway_test calculates empirical p-values for gene-sets.
`method`	A character string determines the method to calculate the pathway scores. Currently, "FAIME" (default), "KS-rank", and "cumulative-rank" are supported.

genome — A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

alpha — A positive integer, 5 by default. This is a FAIME-specific parameter. A higher value puts more weights on the most highly-expressed ranks than the lower expressed ranks.

logCheck — A Boolean value. By default is FALSE. When true, the function takes the log-transformed values of gene if the maximum value of sample profile is larger than 20.

na.rm — A Boolean value indicates whether to keep missing values or not when method="FAIME". By default is FALSE.

B — A positive integer assigns the total number of random sampling trials to calculate the empirical pvalues. By default is 100.

min_Intersect_Count — A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.

Value

A list or data frame. If the parameter "FisherTest" is true, the result is a list including both reports for Fisher's exact test and the gene2pathway test. Otherwise, only reports the gen2pathway tested results.

Author(s)

Xinan Yang

Examples

```
data(dat_chip)
data(MsigDB_C5,package="seq2pathway.data")
#generate a demo GSA.genesets object
demoDB <- MsigDB_C5
x=100
for(i in 1:3) demoDB[[i]]<-MsigDB_C5[[i]][1:x]

res<-gene2pathway_test(dat=head(dat_chip), DataBase=demoDB,
FisherTest=FALSE, EmpiricalTest=FALSE,
        method="FAIME", genome="hg19", min_Intersect_Count=1)
# check ther result
names(res)
res[[1]]
res[[2]]
## Not run:
res<-gene2pathway_test(dat=head(dat_chip), DataBase="BP",
FisherTest=FALSE, EmpiricalTest=FALSE,
        method="FAIME", genome="hg19", min_Intersect_Count=1)

## End(Not run)
```

GRanges_demo *loci information with GRanges format*

Description

loci information with GRanges format

Usage

```
data("GRanges_demo")
```

Format

GRanges object with 10 ranges and 3 metadata columns.

Value

GRanges object

References

Lawrence M, Huber W, Pages H, Aboyoun P, Carlson M, Gentleman R, Morgan M and Carey V(2013). "Software for Computing and Annotating Genomic Ranges." PLoS Computational Biology, 9.

Examples

```
data(GRanges_demo)
```

runseq2gene *R wrapped python function to map genomic regions on the sequence-level to genes.*

Description

Annotate genome regions of interest to either the nearest TSS or a broader range of neighboring genes.

Usage

```
runseq2gene(inputfile,
            search_radius=150000, promoter_radius=200, promoter_radius2=100,
            genome=c("hg38","hg19","mm10","mm9"), adjacent=FALSE, SNP=FALSE,
            PromoterStop=FALSE,NearestTwoDirection=TRUE,UTR3=FALSE)
```

Arguments

`inputfile` An R object input file that records genomic region information (coordinates). The file format could be data frame defined as:

1. column 1 the unique IDs of genomic regions of interest (peaks, mutations, or SNPs)
2. column 2 the chromosome IDs (eg. chr5 or 5)
3. column 3 the start of genomic regions
4. column 4 the end of genomic regions (for SNP and point mutations, the difference of start and end is 1bp)
5. column 5... Other custom defined information (option)

Or, the input format should be RangedData object(from R package IRanges) with value column.

1. column 1: space the chromosome IDs (eg. chr5 or 5)
2. column 2: ranges the ranges of genomic regions
3. column 3: name the unique IDs of genomic regions of interest (peaks, mutations, or SNPs)
4. more columns: Other custom defined information (optional)

`search_radius` A non-negative integer, with which the input genomic regions can be assigned not only to the matched or nearest gene, but also with all genes within a search radius for some genomic region type. This parameter works only when the parameter "SNP" is FALSE. Default is 150000.

`promoter_radius` A non-negative integer. Default is 200. Promoters are here defined as upstream regions of the transcription start sites (TSS). User can assign the promoter radius, a suggested value is between 200 to 2000.

`promoter_radius2` A non-negative integer. Default is 100. Promoters are here defined as downstream regions after the transcription start sites (TSS).

`genome` A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

`adjacent` A Boolean. Default is FALSE to search all genes within the search_radius. Using "TRUE" to find the adjacent genes only and ignore the parameters "SNP" and "search_radius".

`SNP` A Boolean specifies the input object type. FALSE by default to keep on searching for intron and neighboring genes. Otherwise, runseq2gene stops searching when the input genomic region is residing on exon of a coding gene.

`PromoterStop` A Boolean, "FALSE" by default to keep on searching neighboring genes using the parameter "search_radius". Otherwise, runseq2gene stops searching neighboring genes. This parameter has function only if an input genomic region maps to promoter of coding gene(s).

`NearestTwoDirection` A boolean, "TRUE" by default to output the closest left and closest right coding genes with directions. Otherwise, output only the nearest coding gene regardless of direction.

`UTR3` A boolean, "FALSE" by defalt to calculate the distance from genes' 5UTR. Otherwsie, calculate the distance from genes' 3UTR.

Value

A matrix with multiple columns.

`Columns 1 to 4` The same as the first four columns in the input file.

`PeakLength` An integer gives the length of the input genomic region. It is the number of base pairs between the start and end of the region.

`PeakMtoStart_Overlap`
An integer gives the distance from the TSS of mapped gene to the middle of genomic region. A negative value indicates that TSS of the mapped gene is at the right of the peak. Otherwise, PeakMtoStart_Overlap reports a numeric range showing the location of overlapped coordinates (exon, intron, CDS, or UTR).

`type` A character specifies the relationship between the genomic region and the mapped gene.

1. "Exon" any part of a genomic region overlaps the exon region of the mapped gene
2. "Intron" any part of a genomic region overlaps an intron region of the mapped gene
3. "cds" any part of a genomic region overlaps the CDS region
4. "utr" any part of a genomic region overlaps a UTR region
5. "promoter" any part of a genomic region overlaps the promoter region of the mapped gene when an intergenic region of mapped gene covers the input genomic region
6. "promoter_internal" any part of a genomic region overlaps the promoter region of the mapped gene when an adjacent TTS region of mapped gene covers the input genomic region
7. "Nearest" the mapped gene is the nearest gene if the genomic region is located in an intergenic region
8. "L" and "R" show the relative location of mapped genes when the input genomic region resides within a bidirectional region
9. "Neighbor" any mapped gene within the search radius but belongs to none of the prior types

`BidirectionalRegion`
A Boolean indicates whether or not the input genomic region is in bidirectional region. *"A 'bidirectional gene pair' refers to two adjacent genes coded on opposite strands, with their 5' UTRs oriented toward one another." (from wiki http://en.wikipedia.org/wiki/Promoter_(genetics)).* NA means the genomic region is at exon or intron region.

`Chr` An integer gives chromosome number of mapped gene.

`TSS` An integer indicates transcription start site of mapped gene regardless of strand.

`TTS` An integer indicates transcription termination site of mapped gene regardless of strand.

`strand` A character indicates whether mapped gene is in forward (+) or reverse (-) direction on chromosome.

`gene_name` A character gives official gene symbol of mapped genes.

`source` A character gives gene source (Ensembl classification) of mapped genes.

`transID` A character gives Ensemble transcript ID of mapped genes.

Author(s)

Bin Wang

References

Lawrence M, Huber W, Pages H, Aboyoun P, Carlson M, Gentleman R, Morgan M and Carey V (2013) "Software for Computing and Annotating Genomic Ranges.". *PLoS Computational Biology*, **9**.

Examples

```
data(Chipseq_Peak_demo)
res=runseq2gene(inputfile=Chipseq_Peak_demo)
```

`runseq2pathway` *An function to perform the runseq2pathway algorithm(s).*

Description

A wrapper function to perform seq2gene and gene2pathway in series.

Usage

```
runseq2pathway(inputfile,
               search_radius=150000, promoter_radius=200, promoter_radius2=100,
              genome=c("hg38","hg19","mm10","mm9"), adjacent=FALSE, SNP= FALSE,
               PromoterStop=FALSE, NearestTwoDirection=TRUE,UTR3=FALSE,
               DataBase=c("GOterm"), FAIMETest=FALSE, FisherTest=TRUE,
               collapsemethod=c("MaxMean","function","ME",
               "maxRowVariance","MinMean","absMinMean","absMaxMean","Average"),
             alpha=5, logCheck=FALSE, B=100, na.rm=FALSE, min_Intersect_Count=5)
```

Arguments

`inputfile` An R object input file that records genomic region information (coordinates). The file format could be data frame defined as:

1. column 1 the unique IDs of genomic regions of interest (peaks, mutations, or SNPs)
2. column 2 the chromosome IDs (eg. chr5 or 5)
3. column 3 the start of genomic regions
4. column 4 the end of genomic regions (for SNP and point mutations, the difference of start and end is 1bp)
5. column 5... Other custom defined information (option)

Or, the input format should be GRanges object(from R package GenomicRanges) with value column.

1. column 1: space the chromosome IDs (eg. chr5 or 5)
2. column 2: ranges the ranges of genomic regions
3. column 3: name the unique IDs of genomic regions of interest (peaks, mutations, or SNPs)

4. more columns: Other custom defined information (optional)

`search_radius` A non-negative integer, with which the input genomic regions can be assigned not only to the matched or nearest gene, but also with all genes within a search radius for some genomic region type. This parameter works only when the parameter "SNP" is FALSE. Default is 150000.

`promoter_radius` A non-negative integer. Default is 200. Promoters are here defined as upstream regions of the transcription start sites (TSS). User can assign the promoter radius, a suggested value is between 200 to 2000.

`promoter_radius2` A non-negative integer. Default is 100. Promoters are here defined as downstream regions after the transcription start sites (TSS).

`genome` A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

`adjacent` A Boolean. Default is FALSE to search all genes within the search_radius. Using "TRUE" to find the adjacent genes only and ignore the parameters "SNP" and "search_radius".

`SNP` A Boolean specifies the input object type. FALSE by default to keep on searching for intron and neighboring genes. Otherwise, runseq2gene stops searching when the input genomic region is residing on exon of a coding gene.

`PromoterStop` A Boolean, "FALSE" by default to keep on searching neighboring genes using the parameter "search_radius". Otherwise, runseq2gene stops searching neighboring genes. This parameter has function only if an input genomic region maps to promoter of coding gene(s).

`NearestTwoDirection` A boolean, "TRUE" by default to output the closest left and closest right coding genes with directions. Otherwise, output only the nearest coding gene regardless of direction.

`UTR3` A boolean, "FALSE" by defalt to calculate the distance from genes' 5UTR. Otherwsie, calculate the distance from genes' 3UTR.

`DataBase` A character string assigns an R GSA.genesets object to define gene-set. User can call GSA.read.gmt to load customized gene-sets with a .gmt format. If not specified, a character "GOterm" by default, three categories of GO-defined gene sets (BP,MF,CC) will be used. Alternatively, user can specify a category by the choice of "BP","MF","CC".

`FAIMETest` A boolean values. By default is FALSE. When true, executes function of gene2pathway test using the FAIME method, which only functions when the fifth column of input file exsists and is a vector of scores or values.

`FisherTest` A Boolean value. By default is TRUE to excute the function of the Fisher's exact test. Otherwise, only excutes the function of gene2pathway test.

`collapsemethod` A character for determining which method to use when call the function collapseRows in package WGCNA. The function "collapsemethod" uses this paramter to call the collapseRows() function in package "WGCNA".

`alpha` A positive integer, 5 by default. This is a FAIME-specific parameter. A higher value puts more weights on the most highly-expressed ranks than the lower expressed ranks.

`logCheck` A Boolean value. By default is FALSE. When true, the function takes the log-transformed values of gene if the maximum value of sample profile is larger than 20.

`na.rm` A Boolean value indicates whether to keep missing values or not when method="FAIME". By default is FALSE.

`B` A positive integer assigns the total number of random sampling trials to calculate the empirical pvalues. By default is 100.

`min_Intersect_Count` A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.

Value

An R list of several data frames. The results of function seq2gene, Fisher's exact test and gene2pathway test results are included.

Author(s)

Bin Wang, Xinan Yang

References

Langfelder P, Horvath S (2008) WGCNA: an R package for weighted correlation network analysis. *BMC Bioinformatics*, 9:559.

Miller JA, Cai C, Langfelder P, Geschwind DH, Kurian SM, Salomon DR, Horvath S (2011) Strategies for aggregating gene expression data: The collapseRows R function. *BMC Bioinformatics*, 12:322.

Lawrence M, Huber W, Pages H, Aboyoun P, Carlson M, Gentleman R, Morgan M and Carey V (2013) "Software for Computing and Annotating Genomic Ranges.". *PLoS Computational Biology*, **9**.

Examples

```
data(Chipseq_Peak_demo)
require(seq2pathway.data)
data(MsigDB_C5, package="seq2pathway.data")
  #generate a demo GSA.genesets object
demoDB <- MsigDB_C5
x=10
for(i in 1:3) demoDB[[i]]<-MsigDB_C5[[i]][1:x]
       res3=runseq2pathway(inputfile=Chipseq_Peak_demo,
genome="hg19", search_radius=100, promoter_radius=50, promoter_radius2=0,
FAIMETest=TRUE, FisherTest=FALSE,
DataBase=demoDB, min_Intersect_Count=1)
names(res3)
res3[[1]]
  ## Not run:
   # an example to use FET
res=runseq2pathway(inputfile=Chipseq_Peak_demo,
genome="hg19", search_radius=100, promoter_radius=50, promoter_radius2=0,
DataBase=MsigDB_C5, NearestTwoDirection=FALSE,
collapsemethod="Average", min_Intersect_Count=1)
   # an example to use FAIME
res2=runseq2pathway(inputfile=Chipseq_Peak_demo,
genome="hg19", search_radius=100, promoter_radius=50, promoter_radius2=0,
FAIMETest=TRUE, FisherTest=FALSE,
DataBase=MsigDB_C5, min_Intersect_Count=1)
```

```
## End(Not run)
```

Index