Package 'SMITE'

October 12, 2016

Type Package

Title Significance-based Modules Integrating the Transcriptome and Epigenome

Version 1.0.2

Date 2016-01-01

Author Neil Ari Wijetunga, Andrew Damon Johnston, John Murray Greally

Maintainer Neil Ari Wijetunga <Neil.Wijetunga@med.einstein.yu.edu>, Andrew Damon Johnston <Andrew.Johnston@med.einstein.yu.edu>

Description This package builds on the Epimods framework which facilitates finding weighted subnetworks (``modules'') on Illumina Infinium 27k arrays using the SpinGlass algorithm, as implemented in the iGraph package. We have created a class of gene centric annotations associated with p-values and effect sizes and scores from any researchers prior statistical results to find functional modules.

License GPL (>=2)

Depends R (>= 3.3), GenomicRanges

Imports scales, plyr, Hmisc, AnnotationDbi, org.Hs.eg.db, ggplot2, reactome.db, KEGG.db, BioNet, goseq, methods, IRanges, igraph, Biobase,tools, S4Vectors, geneLenDataBase, grDevices, graphics, stats, utils

Suggests knitr

VignetteBuilder knitr

biocViews DifferentialMethylation, DifferentialExpression, SystemsBiology, NetworkEnrichment,GenomeAnnotation,Network, Sequencing, RNASeq, Coverage

URL https://github.com/GreallyLab/SMITE

BugReports https://github.com/GreallyLab/SMITE/issues

NeedsCompilation no

R topics documented:

`SMITE-package` *Significance-based Modules Integrating the Transcriptome and Epigenome*

Description

SMITE provides a method of scoring and visualizing multi-level epigenomic data in order to prioritize genes within a genome-wide experiment. These scores can then be used to identify subnetworks within an interaction network called modules. Each module represents a collection of highly interacting genes that are implicated by the experiment.

Details

Package:	SMITE
Type:	Package
Version:	1.0.0
Date:	2015-07-06
License:	GPL (>=2)

Author(s)

Neil Ari Wijetunga, Andrew Damon Johnston

Maintainer: Neil.Wijetunga@med.einstein.yu.edu, Andrew.Johnston@med.einstein.yu.edu

See Also

FEM BioNet

Examples

```
## NOTE: commented out for example.  See vignette for better explanation ##

options(stringsAsFactors=FALSE)

data(methylationdata)
methylation <- methylation[-which(is.na(methylation[, 5])), ]
methylation[,5] <- replace(methylation[,5],methylation[,5] == 0,
    min(subset(methylation[,5], methylation[,5]!=0), na.rm=TRUE))

data(curated_expressiondata)

data(hg19_genes_bed)
data(histone_h3k4me1)

#test_annotation<-makePvalueAnnotation( data=hg19_genes,
#other_data=list(h3k4me1=h3k4me1), gene_name_col=5, other_tss_distance=5000)

##fill in expression data
#test_annotation<-annotateExpression(test_annotation, expression_curated)


##fill in methylation data

#test_annotation<-annotateModification(test_annotation, methylation,
#weight_by=c(promoter="distance", body="distance", h3k4me1="distance"),
#verbose=TRUE, mod_corr=TRUE)

##create a pvalue object that will count the effect of the h3k4me1 as
##bidirectional

#test_annotation<-makePvalueObject(test_annotation,
```

```
#effect_directions=c(methylation_promoter="decrease",
#methylation_body="decrease",
#methylation_h3k4me1="bidirectional"))

##normalize the pvalues compared to colExp

#test_annotation<-normalizePval(test_annotation,ref="expression_pvalue",
#method="rescale")

##score with all four features contributing

#test_annotation<-SMITEscorePval(test_annotation,
#weights=c(methylation_promoter=.3,methylation_body=.1,expression=.3,
#methylation_h3k4me1=.3))

##load REACTOME
#load(system.file("data","Reactome.Symbol.Igraph.rda", package="SMITE"))

##run Spinglass using REACTOME network

#test_annotation<-runSpinglass(test_annotation, REACTOME, maxsize=50,
#num_iterations=10)

##run goseq on individual modules to determine bias
#test_annotation <- runGOseq(test_annotation,
#coverage=read.table(system.file("extdata",
#"hg19_symbol_hpaii.sites.inbodyand2kbupstream.bed.gz", package="SMITE")),
#type="kegg")

##search go seq output for keywords
#searchGOseq(test_annotation, "Cell")

##Draw a network
#plotModule(test_annotation, which_network=6, layout="fr")

##sample final file ##
data(test_annotation_score_data)
```

---

addShadowText *Add shadow text (a second color bordering the text) to a plot*

Description

This is a usefule function to help text stand out on busy backgrounds like gene networks

Usage

```
addShadowText(x, y = NULL, labels, col = "white", bg = "black",
theta = seq(pi/4, 2 * pi, length.out = 8), r = 0.1, ...)
```

Arguments

x	A numeric vector of x coordinates
y	A numeric vector of y coordinates
labels	A character vector to be plotted at the specified coordinates
col	The text color
bg	The color of the outline
theta	The number of shadows to plot
r	The radius for the shadows
...	Additional plotting arguments

Details

The function creates its effect by plotting theta shadows at r radius around the text to create the illusion of a text shadow

Value

Adds shadow text to plot

Note

This function was adapted by N. Ari Wijetunga for SMITE.

Author(s)

Greg.Snow <at> imail.org

References

http://article.gmane.org/gmane.comp.lang.r.general/147787

See Also

text, mtext

Examples

```
plot.new()

addShadowText(x = .5,y = .5,"TEST",col="white",bg="gray")
```

annotateExpression *Adding expression data to a PvalueAnnotation*

Description

This function is used to create and load an ExpressionSet into a PvalueAnnotation. Using specified effect and p-value column or named columns that the function will use to determine the effect and p-value columns, it loads the data it into the PvalueAnnotation.

Usage

```
annotateExpression(pvalue_annotation, expr_data, effect_col = NULL, pval_col = NULL)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation
expr_data	An object of class data.frame or matrix with row names corresponding to genes and atleast two columns with an effect and p-value for expression.
effect_col	A numeric specifying the column with an effect direction. If not specified the function will grep for a single named column from: "effect", "odds", "coeff" or "B"
pval_col	A numeric specifying the column with p-values. If not specified the function will grep for a single named column from: "pval","p.val","p_val" or "sig"

Details

The function will load the entire given expression dataset as an ExpressionSet in the expression slot,while the effect and p-value data will also be stored as an "AnnotatedDataFrame" in the phenoData slot of the ExpressionSet.

Value

A PvalueAnnotation, an S4 object with the slot "expression" filled in.

Author(s)

N. Ari Wijetunga

See Also

annotateModification makePvalueAnnotation createPvalueObject

Examples

```
data(curated_expressiondata)
data(test_annotation_score_data)
## Load Expression data into PvalueAnnotation ##
test_annotation <- annotateExpression(pvalue_annotation=test_annotation,
expression_curated)

## Extract entire ExpressionSet with expression data ##
#slot(test_annotation,"expression")

## Extract expression data summary ##
#head(extractExpression(pvalue_annotation=test_annotation))
```

---

annotateModification *Adding modification data to a PvalueAnnotation*

Description

This function is the main "workhorse" function for SMITE because given a specific epigenetic modification (e.g. DNA methylation) it will 1) assess an internal correlation structure and 2) aggregate the modification over all intervals associated with a gene in the "makePvalueAnnotation" function.

Usage

```
annotateModification(pvalue_annotation, mod_data, weight_by = NULL,
weight_by_method = "Stouffer", mod_included = NULL, mod_corr = TRUE,
mod_type = "methylation", verbose = FALSE)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation.
mod_data	A dataframe or matrix derived from a bed file with thhe the first three columns as (chromosome, start, end), column 4 is the effect, and column 5 is the p-value.
weight_by	A vector with named elements specifying how modifications should be weighted within an interval. Must be one of: "distance" Use the distance from the gene TSS to weight the p-values and the combined effect such that events closer to the TSS are weighted more. Log distances are used. "pval" "p.value" "pvalue" "p_val" (DEFAULT) Do not weight p-values but weight the combined effect such by the signficance of that effect. ELSE Do not weight p-values or the combined effect.

`weight_by_method`
: A character specifying which method should be used to combine p-values. Must be one of:

  "Stouffer" (DEFAULT) Stouffer's method for combing pvalues involves first taking the inverse standard normal CDF transformation of a vector of p-values followed by a weighted sum creating a new Z score with a standard normal distribution "Fisher" "fisher" "chisq" "chi" Fisher's method involves summing the -2ln(p) for each of k p's which follows an approximate chi square distribution with 2k degrees of freedom "Sidak" "sidak" "minimum" Sidak's adjustment is essentially the minimum p-value, with an added transformation to account for multiple comparisons. "binomial" The binomial probability assesses the probability of observing the observed number of p-value below a threshold (alpha=0.05) given the total number of p values and the probability of a false positive.

`mod_included`
: A vector of named elements specifying for which intervals in the annotation the function should find combined scores (e.g. promoters). If not specified the assumption is that all type of intervals associated with a gene should be included.

`mod_corr`
: A logical (TRUE/FALSE) specifying whether a correlation matrix should be estimated. The DEFAULT is TRUE.

`mod_type`
: A character naming the modification that is being loaded. The DEFAULT is "methylation" and any modType string can be used, but will be referred to in downstream analysis. A unique name must be used for each modification that is loaded. When picking a variable modType should also avoid using "_" as it is used to split column names containing modType.

`verbose`
: A logical specifying if the user wants updates about the progress of the function.

Details

This function is the main "workhorse" function for SMITE because given a specific epigenetic modification (e.g. DNA methylation) it will 1) assess an internal correlation structure and 2) aggregate the modification over all intervals associated with a gene in the "makePvalueAnnotation" function.

Value

A an S4 object of class PvalueAnnotation with the slot modification (a GrangesList) filled in for each additional modification.

Author(s)

N. Ari Wijetunga

References

Fisher R. Statistical methods for research workers. Oliver and Boyd; Edinburgh: 1932.

Stouffer S, DeVinney L, Suchmen E. The American soldier: Adjustment during army life. Vol. 1. Princeton University Press; Princeton, US: 1949.

Sidak, Z. (1967). Rectangular confidence regions for the means of multivariate normal distributions, Journal of the American Statistical Association 62, 626 633.

See Also

removeModification annotateExpression makePvalueAnnotation createPvalueObject

Examples

```
options(stringsAsFactors=FALSE)

## Commented out below See vignette for more detailed usage information ##
## Load genome bed file ##
#data(hg19_genes_bed)

## Create a PvalueAnnotation with defaults for promoter size##
#test_annotation<-makePvalueAnnotation(data=hg19_genes, gene_name_col=5)

## Load DNA methylation bed file ##
#data(methylationdata)
#methylation<-methylation[-which(is.na(methylation[,5])),]
#methylation[,5]<-replace(methylation[,5],methylation[, 5] == 0,
#min(subset(methylation[,5], methylation[,5]!=0), na.rm=TRUE))

## Load DNA methylation into PvalueAnnotation modCorr=F for example##
## NOTE: Commented out below.  See vignette for better example ##
#test_annotation <- annotateModification(pvalue_annotation=test_annotation,
#mod_data=methylation, weight_by=c(promoter="distance", body="distance"),
#verbose=FALSE, mod_corr=FALSE, mod_type="methylation")
```

---

convertGeneIds *Convert between gene ids*

Description

A convenient function used to convert between gene ids from different gene annotations.

Usage

```
convertGeneIds(gene_IDs, ID_type, ID_convert_to, delim = NULL, verbose = FALSE)
```

Arguments

gene_IDs	A vector of gene names.
ID_type	A character specifying the type of given annotation. Currently one of "refseq", "ensembleprot","uniprot" or "ensemble". In the case of ID_convert_to="entrez", "symbol"
ID_convert_to	A character specifying the type of desired annotation. Currently one of "symbol" or in they case of ID_type="symbol", "entrez"
delim	An optional character that will be removed from the beginning of each gene name. It can be a long string.
verbose	TRUE/FALSE Should the function be verbose? DEFAULTS to FALSE.

Details

This is a very usefule function to efficently convert between gene ids. It currently relies on enumeration of each possible conversion, which has limited it's use to mainly converting to gene symbol.

Value

A character vector formatted to ID_convert_to

Note

The function has enumerated combinations using AnnotationDBI. We can provide additional functionality if needed.

Author(s)

N. Ari Wijetunga < Neil.Wijetunga@med.einstein.yu.edu >

See Also

AnnotationDBI, Biomart

Examples

```
data(genes_for_conversiontest)

genes[,1] <- convertGeneIds(gene_IDs=genes[,1], ID_type="refseq",
                            ID_convert_to="symbol")
```

---

curated_expressiondata

A toy dataset ofcurated RNA-seq to test within SMITE

Description

A toy dataset of pre-cleaned gene expression data from RNA-seq. The file is effect and p-value with gene names as rownames.

Usage

```
data("curated_expressiondata")
```

Format

A data frame with 20819 observations on the following 2 variables.

rownames a character vector specifying gene

column1 an numeric vector specifying effect (log fold change)

column2 a numeric vector with a two sided p-value from DESeq analysis

Details

This gene expression dataset is a randomized version of the Toxoplasma dataset used to benchmark SMITE. It no longer has NAs or p-values=0. Gene names were converted to gene symbols.

Value

A dataframe with rownames as genes in Refseq format and columns for effects and pvalues derived from negative binomial testing of DESeq normalized values from RNA-seq.

Source

Manuscript in preparation. Please see https://github.com/GreallyLab for more details.

Examples

```
data(curated_expressiondata)
```

---

extractExpression *View the expression data stored in a PvalueAnnotation*

Description

This function allows the user to see the effect and p-value data that was loaded into a PvalueAnnotation before performing downstream analysis.

Usage

```
extractExpression(pvalue_annotation)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation for which expression data has already been loaded via annotateExpression

Value

A data.frame pulled from the phenoData of the expression slot within a load PvalueAnnotation. The phenoData specifically hold the effect and p-value information.

Author(s)

N. Ari Wijetunga

See Also

annotateExpression

Examples

```
data(test_annotation_score_data)
data(curated_expressiondata)
## Load Expression data into PvalueAnnotation ##
test_annotation<-annotateExpression(test_annotation, expression_curated)

## Extract entire ExpressionSet with expression data ##
#slot(test_annotation,"expression")

## Extract expression data summary ##
head(extractExpression(pvalue_annotation=test_annotation))
```

`extractGOseq` *View the GOseq pathway analysis after having run Goseq, or search for a term.*

Description

Having defined at least one genomic module using runSpinglass or runBioNet, this function allows you to interrogate the enriched terms for a specific module or combination of modules.

Usage

```
extractGOseq(pvalue_annotation, which_network = NULL)
searchGOseq(pvalue_annotation, search_string, wholeword = FALSE)
```

Arguments

`pvalue_annotation`	An S4 object of class PvalueAnnotation, for which module-finding and GOseq analysis have already been performed
`which_network`	A numeric vector of a length of at least one, corresponding to a particular functional module specifically for the extract function.
`search_string`	A character specfying a search string specifically for the search function.
`wholeword`	A logical (TRUE/FALSE) detemring whether the search string must be matched for whole word specifically for the search function.

Details

Goseq analysis is useful since it allows you to assess term/pathway enrichment in a collection of genes, while adjusting for bias data. Potential bias can be from aspects like gene length or probe density that influence the likelihood of finding a particular gene. For more information please see the goseq reference.

Value

##Extract## A list with eahc element matching the specified module. Has columns identfying the term id, the over represented p-value, underrepresented p-value the total number in the category found in the module, the total number in the category and a more descirptive pathway name.

##Search## A matrix with columns identfying the module name, module position/significance , the specific enriched term, the rank of that term within all enriched terms and the total number of enriched terms.

Author(s)

N. Ari Wijetunga

References

Young MD, Wakefield MJ, Smyth GK and Oshlack A (2010). Gene ontology analysis for RNA-seq: accounting for selection bias. Genome Biology, 11, pp. R14.

See Also

runGOseq runSpinglass runBioNet extractModules plotModule

Examples

```
## Commented out below.  See vignette for more details ##

##load sample data with only PvalueObject filled in##
data(test_annotation_score_data)

## show goseq analysis for module 1 ##
extractGOseq(test_annotation, 1)

## show goseq analysis for module 1 and 2 ##
#extractGOseq(test_annotation, 1:2)

## search for term ##

#searchGOseq(test_annotation, "Cell cycle")
```

`extractModification` *Extract some or all loaded modifications or a the summary of combined effects*

Description

After having loaded modifications into a PvalueAnnotation, these functions can be used to display the GRanges with the modification of interest, or the data frame containg a summary of the combined effects.

Usage

```
extractModification(pvalue_annotation, mod_type = "methylation")
extractModSummary(pvalue_annotation)
```

Arguments

pvalue_annotation	An s4 object of calss PvalueAnnotation
mod_type	A string or character vector that must match one or more of the loaded modifications. If NULL (DEFAULT) then it will show all modifications.

Value

A GRanges object containing the modification(s) of interest or a data frame with a summary of the combined p-values and effects

Author(s)

N. Ari Wijetunga

See Also

extractExpression annotateModificaiton removeModification

Examples

```
##NOTE: Comment out in exmaple see vignette for more detailed usage ##

## Load genome bed file ##
data(hg19_genes_bed)

## Load curated DNA methylation bed file ##
#data(methylationdata)
#methylation <- methylation[-which(is.na(methylation[,5])),]
#methylation[, 5] <- replace(methylation[,5],methylation[,5] == 0,
#min(subset(methylation[, 5], methylation[, 5] !=0 ), na.rm=TRUE))

## Create a PvalueAnnotation with defaults for promoter size##
test_annotation<-makePvalueAnnotation(data=hg19_genes, gene_name_col=5)

## Load DNA methylation into PvalueAnnotation ##
#test_annotation <- annotateModification(pvalue_annotation=test_annotation,
#methylation, weight_by=c(promoter="distance", body="distance"), verbose=TRUE,
#mod_corr=FALSE, mod_type="methylation")

## Extract GRanges with modification data ##
#extractModification(pvalue_annotation=test_annotation)
```

`extractModules` *View specific modules within a PvalueAnnotation*

Description

Having identfied modules within a Pvalue annotation, this function allows the user to display 1 or more of the module genes.

Usage

```
extractModules(pvalue_annotation, which_module = NULL)
```

Arguments

`pvalue_annotation`	An S4 object of class PvalueAnnotation for which Spinglass or BioNet has already been run.
`which_module`	A numeric vector specifying one or more module to display

Value

A list with each element containing the requested modules

Author(s)

N. Ari Wijetunga

See Also

plotModule runGOseq extractGOseq runSpinglass runBioNet

Examples

```
data(test_annotation_score_data)

extractModules(pvalue_annotation=test_annotation, which_module=1)
```

`extractScores` *Extract scores for all genes*

Description

A function to obtain all gene scores

Usage

```
extractScores(pvalue_annotation)
```

Arguments

`pvalue_annotation`
: An S4 object of class PvalueAnnotation for which scores have already been calculated

Value

A named vector containing all gene scores

Author(s)

N. Ari Wijetunga

See Also

scorePval extractModules highScores

Examples

```
data(test_annotation_score_data)

out <- extractScores(pvalue_annotation=test_annotation)

head(out)
```

genes_for_conversiontest *A small set of RefSeq genes for converting*

Description

This toy dataset has 100 randomly selected RefSeq genes and can be used to test conversion functionality in SMITE

Usage

```
data("genes_for_conversiontest")
```

Format

A data frame with 100 observations on the following 1 variables.

column1 a character vector of RefSeqGene IDs

Value

A dataframe with genes in Refseq format for conversion testing.

Examples

```
data("genes_for_conversiontest")

genes[,1]<-convertGeneIds(gene_IDs=genes[,1], ID_type="refseq",
                          ID_convert_to="symbol")
```

hg19_genes_bed *A bed file annotating Refseq genes for the hg19 genome build*

Description

A gene anntation BED file containing columns for RefSeq name and Gene Symbol

Usage

```
data("hg19_genes_bed")
```

Format

A data frame with 41633 observations on the following 6 variables.

column 1 a character vector for chromosome

column 2 an integer vector for start position

column 3 an integer vector for end position

column 4 an character vector for RefSeq gene names

column 5 an character vector for Gene Symbol names

column 6 an character vector for strand

Details

A BED files taken from the table browser.

Value

A dataframe in BED format (chromosome, start, end) with additional columns for gene name as Refseq and gene symbol and strand.

Source

Karolchik D, Hinrichs AS, Furey TS, Roskin KM, Sugnet CW, Haussler D, Kent WJ. The UCSC Table Browser data retrieval tool. Nucleic Acids Res. 2004 Jan 1;32 (Database issue):D493-6.

References

http://genome.ucsc.edu/

Examples

```
data(hg19_genes_bed)
```

---

highScores *Generate a vector of the highest scoring genes*

Description

This function can be used to extract a subset of the highest scoring genes for other downstream analysis.

Usage

```
highScores(pvalue_annotation, alpha = 0.05)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation for which scoring has already been performed
alpha	A numeric specifying a threshold at signficant genes can be determined. DEFAULT is alpha=0.05.

Details

This function randomly samples the scores with replacement 100 times and for within each random sample for each score it determines the proportion of scores at or greater than the score. The average of these proportions over the 100 samples will be the new p-value/scores. All scores falling below the threshold will be returned.

Value

A named vector of scores.

Author(s)

N. Ari Wijetunga

See Also

scorePval plotCompareScores runSpinglass runBioNet

Examples

```
data(test_annotation_score_data)


## Note: commented out for example.  See vignette for better example ##

#out <- highScores(pvalue_annotation=test_annotation, alpha=0.01)
```

---

histone_h3k4me1 *A toy dataset of H3k4me1 peaks to test within SMITE*

Description

A toy dataset of H3k4me1 peaks from liver ChIP-seq through the encode project. The file is a BED file.

Usage

```
data(histone_h3k4me1)
```

Format

A data frame with 75448 observations on the following 3 variables.

column1 a character vector specifying chromsome

column2 an integer vector specifying start

column3 an integer vector specifying end

Details

This is a BED file that specifices the consensus locations of three H3K4me1 ChIP-seq experiments performed on normal adult liver.

Value

A dataframe in BED format (chromosome, start, end).

Source

GSM669972, GSM621654, GSM537706

Roadmap Epigenomics Lister R, et al. Nature. 2009 Nov 19;462(7271):315-22 ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. Nature. 2012 Sep 6;489(7414):57-74.

Examples

```
data(histone_h3k4me1)
head(h3k4me1)
```

---

makePvalueAnnotation *Initialize a PvalueAnnotation*

Description

This function initializes a PvalueAnnotation using a gene BED file and optional BED files corresponding to interval datasets. This is a necessary first step in order to establish for each gene the gene promoter, body and associated intervals.

Usage

```
makePvalueAnnotation(data, other_data = NULL, other_tss_distance = 10000,
    promoter_upstream_distance = 1000, promoter_downstream_distance = 1000,
    strand_col = NULL, gene_name_col = NULL)
```

Arguments

data	A required gene annotation BED file like that obtained from the UCSC Table Browser. At a minimum BED files must have the first three columns as (chromosome, start, end). Additional required columns should correspond to the strand and gene name.The gene name needs to match the gene format desired for the interaction network. Duplicated gene names and associated gene annotations are removed.
other_data	A list of BED files corresponding to each additional interval file to be associated with genes. The function will use the other_tss_distance variable and the gene transcription start site (TSS) to find for each gene all intervals within [tss-other_tss_distance, tss+other_tss_distance].
other_tss_distance	A vector specifying for each element of otherdata a distance from the gene TSS to consider that interval as related to a gene. If the length of other_tss_distance does not match the length of the otherdata list, then the first value is used for all datasets in the otherdata list. DEFAULTS to 10,000 base pairs.
promoter_upstream_distance	A numeric specifying how far upstream from the gene TSS is considered part of the gene promoter. DEFAULTS to 1,000 base pairs.
promoter_downstream_distance	A numeric specifying how far downstream from the gene TSS is considered part of the gene promoter. Gene bodies subtract the promoter_downstream region. DEFAULTS to 1,000 base pairs.
strand_col	A numeric specifying the column of the gene BED file (data) corresponding to the gene strand. If this is not provided, the function will attempt to determine the strand.
gene_name_col	A numeric specifying the column of the gene BED file (data) corresponding to the gene name.

Details

The required only input file is the gene annotation BED file that should have (as all BED files) the chromosome, start and end in columns 1,2 and 3, respectively. Also, there should be a column for gene name and gene strand. The user needs to determine distance from the gene transcription start site that will define the gene promoter. The gene body will then be calculated as the non-promoter overlapping sequence. If optional BED files are given as otherdata (e.g. transcription factor binding sites, histone modification peaks), then the user will also decide a distance from the gene TSS to associate each BED interval with a gene. For a particular BED file, each genes may have more than one interval that falls within the desired range around a TSS. Unique gene names are required and the function will automatically remove duplicated genes. We recommend deciding on an interaction network first and then loading a gene annotation BED file with the same names. This will likely necessitate allowing the function to pick one annotation of a gene, or pre- processing using some criteria (e.g. longest transcript).

Value

An S4 object of class PvalueAnnotation containing slots for an annotation (GRangesList), an expression set, modifications (GRangesList), and a PvalueObject.

Author(s)

N. Ari Wijetunga

See Also

SMITE vignette

Examples

```
## Note: Commented out below. See vignette for more detailed usage information##

## Load genome bed file ##
data(hg19_genes_bed)

## Create a PvalueAnnotation with defaults for promoter size##
test_annotation <- makePvalueAnnotation(data=hg19_genes, gene_name_col=5)
```

---

makePvalueObject *Function to make a PvalueObject within a PvalueAnnotation*

Description

Having annotated modifications and expression data this function will assemble a PvalueObject within the slot "score_data" of a PvalueAnnotation. This is a necessary step before being able to run downstrem functions.

Usage

```
makePvalueObject(pvalue_annotation, effect_directions = NULL)
```

Arguments

`pvalue_annotation`
: An S4 object of class PvalueAnnotation

`effect_directions`
: A character vector with optional names specifying "increase" Modification is expected to increase as expression increase "decrease" Modification is expected to descrease as expression decreases "bidirectional" No direction is assumed between modification and direction

Details

The specified relationship between the modification and expression will be stored and then used when scoring.

Value

An S4 object of class PvalueAnnotation with a slot for score_data filled it

Author(s)

N.Ari Wijetunga

See Also

makePvalueAnnotation

Examples

```
#NOTE: Commented out in example, please see vignette for more details##
options(stringsAsFactors=FALSE)

data(methylationdata)
methylation <- methylation[-which(is.na(methylation[, 5])), ]
#methylation[, 5] <- replace(methylation[, 5],methylation[, 5] == 0,
#min(subset(methylation[, 5], methylation[, 5] != 0), na.rm=TRUE))

#data(curated_expressiondata)
#data(hg19_genes_bed)
#data(histone_h3k4me1)

#test_annotation<-makePvalueAnnotation(data=hg19_genes,
#other_data=list(h3k4me1=h3k4me1), gene_name_col=5,other_tss_distance=5000)

#fill in expression data
#test_annotation<-annotateExpression(test_annotation, expression_curated)


#fill in methylation data
#this step takes ~10 minutes
#test_annotation<-annotateModification(test_annotation, methylation,
#weight_by=c(promoter="distance",body="distance",h3k4me1="distance"),
#verbose=TRUE, mod_corr=FALSE)

#create a pvalue object that will count the effect of the h3k4me1 as
#bidirectional
#test_annotation<-makePvalueObject(pvalue_annotation=test_annotation,
#effect_directions=c(methylation_promoter="decrease",
#methylation_body="decrease", methylation_h3k4me1="bidirectional"))
```

---

methylationdata *A toy dataset of DNA methylation to test within SMITE*

Description

A toy dataset of raw DNA methylation from HELP-tagging. The file is a BED file with columns added for effect and p-value.

Usage

```
data(methylationdata)
```

Format

A data frame with 40000 observations on the following 5 variables.

column1 a character vector specifying chromsome

column2 an integer vector specifying start

column3 an integer vector specifying end

column4 a numeric vector with an effect direction (here it is average difference between two groups)

column5 a numeric vector with a two sided t-test p-value

Details

This is a small subset of a DNA methylation dataset is a randomized version of the Toxoplasma dataset used to benchmark. We could not include the larger version do to package size requirements but larger versions are available. See Github source below. It still has NAs and p-values=0.

Value

A dataframe in BED format (chromosome, start, end) with additional columns for and effect direction and p-value derived from T-tests of HELP-tagging DNA methylation data.

Source

Manuscript in preparation. Please see https://github.com/GreallyLab/SMITE for more details.

Examples

```
data(methylationdata)

any(is.na(methylation[, 4]))

any(methylation[, 4] == 0)
```

---

normalizePval *This function normalizes p-values (Scores) that are otherwise on different scales.*

Description

This function is a used to rescale compenent scores when distributions have been altered. There are two methods available.

Usage

```
normalizePval(pvalue_annotation, trans, ref = "expression_pvalue", method = "rescale")
```

Arguments

pvalue_annotation	An S4 object of class p-value annotation
trans	A vector specifying a specific Box-cox power transformation to use. If not specified, the optimal transformation powers will be estimated.
ref	A string that will be grepped from the names of the loaded expression data or modification/context pairing. All scores will be rescaled to match this reference's range. The DEFAULT is expression.
method	A string of either "Rescale" DEFAULT "rescale" Performs a logit transform and rescales all probabilites to the reference's logit transformed scale, then back-transforms "Box-Cox" "box-cox" "boxcox" "Boxcox" For each probility vector does a logit transform and then iterates between 0.5 and 0.95 by 0.05 to determine the most similar transformation to the logit transformed referece by a Wilcoxon- test

Details

Normalization may not be necessary but should improve some p-values from driving the majority of downstream scores and modules solely because of the scale of their p-values. All transformations are monotonic and are controlled for by use of randomization prodecure downstream. procedures downstream should

Value

An S4 object of class PvalueAnnotation with normalized p-values withing the pval_data slot of the PvalueObject "score_data" slot

Plots densities of p-values before and after transform

Author(s)

N. Ari Wijetunga

See Also

makePvalueObject scorePval plotDensityPval

Examples

```
data(test_annotation_score_data)

#test_annotation<-normalizePval(pvalue_annotation=test_annotation)
```

`plotCompareScores` *Compare two genomic features by score and display them in a hexbin plot*

Description

This function creates a hexbin of the log transformed p-value/score for any two expression or modification-context pairing within a PvalueObject inside of a PvalueAnnotation

Usage

```
plotCompareScores(pvalue_annotation, x_name, y_name, ...)
```

Arguments

`pvalue_annotation`	An S4 object of class PvalueAnnotation for which a PvalueObject has already been created
`x_name`	A string to be grepped from the columns within the slot "pval_data" that is within the PvalueAnnotation slot "score_data." This column will be plotted on the x-axis with a direction specified from the corresponding effect column.
`y_name`	A string to be grepped from the columns within the slot "pval_data" that is within the PvalueAnnotation slot "score_data." This column will be plotted on the y-axis with a direction specified from the corresponding effect column.
`...`	Other plotting parameters

Details

This plotting function creates a hexbin plot of any two p-value vectors stored within a p-value object. It can be used to define relationships between direction and significance in different genomic contexts after having combined p-values.

Value

A hexbin plot

Author(s)

N. Ari Wijetunga

See Also

makePvalueObject plotDensityPval

Examples

```
data(test_annotation_score_data)

plotCompareScores(pvalue_annotation=test_annotation, x_name="expression",
y_name="methylation_promoter")
```

`plotDensityPval` *Plot the density of the combined scores stored in a PvalueObject*

Description

This function in called by the normalizePval function, but can also be called by the user to visualize the relative densities of combined p-values (scores).

Usage

```
plotDensityPval(pvalue_annotation, ref = "expression_pvalue", ...)
```

Arguments

`pvalue_annotation`	An S4 object of class PvalueAnnotation.
`ref`	A character specifying the name of the reference category. DEFAULT is "expression_pvalue"
`...`	Additional plotting arguments

Value

Plots a multidensity plot.

Author(s)

N. Ari Wijetunga

Examples

```
## Load test annotation with only score data ##
data(test_annotation_score_data)

plotDensityPval(pvalue_annotation=test_annotation)
```

plotModule *Plot a specific module after running Spinglass algorithm*

Description

This function is an adapted version of renderModule available through Epimods. We have added optional functionality including plotting the actual raw data onto the node edges, adding goseq annotation to the plot, legends and plotting modes.

Usage

```
plotModule(pvalue_annotation, p_thresh = 0.05, which_network = 1, goseq = FALSE,
layout = "fr", legend = TRUE, namestyle = "symbol", suppress_details = FALSE,
meth_hi_col = "blue", meth_low_col = "yellow1",
meth_mid_col = "gray90", exp_hi_col = "red1", exp_low_col = "chartreuse1",
exp_mid_col = "gray90", label_scale = TRUE, compare_plot=FALSE,
pdf_out=NULL)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation
p_thresh	A numeric specifying a threshold for plotting raw data on edges of nodes. DEFAULT is alpha=0.05. Items above this threshold will be classified as "mid" instead of "high" or "low" "
which_network	A numeric specifying which network to plot. DEFAULTS to 1, and will not plot another network until specified explicitly.
goseq	A logical indicating whether to plot goseq results for the module on the right hand side of the plot.
layout	A character string as either "fr" (DEFAULT) for fruchterman.reingold or "circle" for a circular plot.
legend	A logical (TRUE(DEFAULT)/FALSE) specifying whether a legend should be drawn.
namestyle	A character string as either "symbol" (DEFAULT) for gene symbols or "refseq" for RefSeq genes. If modules were performed on RefSeq genes, then the function will plot with gene symbols so that it is more useful.
suppress_details	A logical (TRUE(DEFAULT)/FALSE) indicating whether border raw data information should be plotted.
meth_hi_col	A color to be associated with signficant modification data with positive effects
meth_low_col	A color to be associated with signficant modification data with negative effects
meth_mid_col	A color to be associated with non-signficant modification data
exp_hi_col	A color to be associated with signficant expression data with postive effects

exp_low_col	A color to be associated with signficant expression data with negative effects
exp_mid_col	A color to be associated with non-signficant expression data
compare_plot	A logical (TRUE/FALSE(DEFAULT)) indicating whether two plots should be drawn side by side, one with raw data and one without
label_scale	A logical (TRUE/FALSE(DEFAULT)) indicating whether whether the node label should be scaled with the node score
pdf_out	A string indicating a location to which the function should output a pdf. If NULL (DEFAULT) then no pdf is made.

Value

A plot of the module

Author(s)

N. Ari Wijetunga

See Also

extractModules

Examples

```
data(test_annotation_score_data)

#plotModule(pvalue_annotation=test_annotation, which_network=2)

#plotModule(pvalue_annotation=test_annotation, which_network=2,
#suppressDetails=TRUE)
```

Reactome.Symbol.Igraph

An Igraph network for REACTOME with nodes as gene symbols

Description

This is an Igraph network that was created using the REACTOME protein-protein interaction database.

Usage

```
data("Reactome.Symbol.Igraph")
```

Format

An igraph object with 5770 nodes and 114288 edges

nodes gene names as gene symbols

edges paired genes that interact

Details

The provided igraph file was created using the igraph package and the interaction file provided from the reference.

Value

An Igraph network based off of REACTOME interactions

Source

REACTOME

References

http://www.reactome.org/pages/download-data/

Examples

```
data(Reactome.Symbol.Igraph)
head(igraph::V(REACTOME))
```

---

removeModification *A function to "unload" a modification that has already been added.*

Description

After using the annotateModification function to load a modification into a PValue annotation, you may wish to remove a modification or reannotate one, which requires removing it first.

Usage

```
removeModification(pvalue_annotation, mod_type = "methylation")
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation
mod_type	A character string that identifies a type of modification within a PvalueAnnotation.

Value

An S4 object of class PvalueAnnotation

Author(s)

N. Ari Wijetunga

See Also

annotateModification extractModification extractModSummary

Examples

```
##NOTE: Commented out in example. ##
## Please see vignette for more detailed usage information ##

## Load genome bed file ##

#data(hg19_genes_bed)

## Load curated DNA methylation bed file ##

data(methylationdata)
methylation <- methylation[-which(is.na(methylation[, 5])), ]
methylation[, 5] <- replace(methylation[, 5], methylation[, 5] == 0,
    min(subset(methylation[, 5], methylation[, 5] != 0), na.rm=TRUE))

#meth1<-methylation

## make second curated test methylation bed file ##

#meth2<-methylation

## Create a PvalueAnnotation with defaults for promoter size##
#test_annotation<-makePvalueAnnotation(data=hg19_genes, gene_name_col=5)

## Load DNA methylation into PvalueAnnotation ##
#test_annotation<-annotateModification(annotation=test_annotation,
#mod_data=meth1, weight_by=c(promoter="distance",body="distance"),verbose=TRUE,
#mod_corr=TRUE,mod_type="methylation")

## Extract GRanges with modification data ##
#extractModification(test_annotation)

## Load second dataset bed file ##
#test_annotation<-annotateModification(pvalue_annotation=test_annotation,
#mod_data=meth2, weight_by=c(promoter="distance",body="distance"),
#verbose=TRUE, mod_corr=TRUE,mod_type="hydroxy")

## Extract GRanges with both modification dataset loaded ##
#head(extractModification(test_annotation,"hydroxy"))

## Unload DNA hydroxymethylation form PvalueAnnotation ##

#test_annotation<-removeModification(pvalue_annotation=test_annotation,
                                    #mod_type="hydroxy")

## Extract GRanges to see only one modification dataset loaded ##
#head(extractModification(pvalue_annotation=test_annotation))
```

runBioNet *Perform BioNet Analysis on a PvalueAnnotation*

Description

With BioNet, a researcher can find a single interconnected gene module using the highest scoring genes generated in a PvalueAnnotation. This function will load the module into the PvalueAnnotation for visualization and downstream analysis.

Usage

```
runBioNet(pvalue_annotation, network, alpha = 0.05)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation that has already had scores generated.
network	An interaction network of class graphNEL or igraph.
alpha	A numeric specfiying a cutoff for high scoring genes to be return with the highScores function.

Details

The input of p-values to BioNet discussed in the BioNet vignette involves first modeling p-values as a Beta-uniform mixture model to obtain the actual corresponding probability function values. Since our scoring method produces p-values/scores that are uniform in distribution, we input them directly into the BioNet algorithm. For more details on BioNet see the reference or runFastHeinz in the BioNet package.

Value

A PvalueAnnotation with a loaded module.

Note

This is a wrapper function to run BioNet. The actual BioNet code was created by Beisser et al.

Author(s)

N. Ari Wijetunga

References

Beisser et al. BioNet: an R-Package for the functional analysis of biological networks. Bioinformatics. 2010 Apr 15;26(8):1129-30. doi: 10.1093/bioinformatics/btq089. Epub 2010 Feb 25.

See Also

plotModule extractModule runGOseq

Examples

```
## load test data ##
data(test_annotation_score_data)


## NOTE: commented out for example.  See vignette for better explanation ##

#load reactome network with gene symbols ##
#load(system.file("data","Reactome.Symbol.Igraph.rda", package="SMITE"))

## run BioNet ##
#test_annotation<-runBioNet(pvalue_annotation=test_annotation,
#network = REACTOME)

## view module ##
#extractModules(pvalue_annotation=test_annotation, 1)

## plot module ##
#plotModule(pvalue_annotation=test_annotation, which.network=1)
```

---

runGOseq *Run a GoSeq pathway analysis*

Description

This function allows pathway annotation of identified modules.

Usage

```
runGOseq(pvalue_annotation, p_thresh = 0.05, coverage, type = "reactome")
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation, for which module-finding has already been performed
p_thresh	A numeric specifying a threshold for signficance of FDR (q-values). DEFAULT is alpha=0.05
coverage	A data.frame that is a bed file (chr start stop) folowed by a gene name and a numeric specifying the bias data (e.g. gene length or number of probes related to gene)
type	Either "kegg" to run KEGG analysis or "reactome" to run a REACTOME analysis

Details

Goseq analysis is useful since it allows you to assess term/pathway enrichment in a collection of genes, while adjusting for bias data. Potential bias can be from aspects like gene length or probe density that influence the likelihood of finding a particular gene. For more information please see the goseq reference.

The function will compare all of the genes within a module to known pathways and terms to find the terms that are most enriched within a module. In this way, this tool allows a reasearch to find a functional importance of a module.

We currently offer KEGG and REACTOME analysis, although additional pathway tools may be added in the near future.

Value

A PvalueAnnotation with goseq annotated modules.

Note

This is a wrapper function written by N. Ari Wijetunga for the package SMITE.

Author(s)

Matthew D. Young myoung at wehi.edu.au

References

Young MD, Wakefield MJ, Smyth GK and Oshlack A (2010). Gene ontology analysis for RNA-seq: accounting for selection bias. Genome Biology, 11, pp. R14.

See Also

searchGOseq extractGOseq runSpinglass runBioNet extractModules plotModule

Examples

```
##load sample data with only PvalueObject filled in##
data(test_annotation_score_data)

## NOTE commented out in example.  Please see Vignette for better example ##
#test_annotation<-runGOseq(pvalue_annotation=test_annotation,
#coverage=read.table(
#system.file("extdata", "hg19_symbol_hpaii.sites.inbodyand2kbupstream.bed.gz",
#package="SMITE"),stringsAsFactors=FALSE),type="kegg")


## search for a term ##
searchGOseq(test_annotation,"Cell cycle")

## show goseq analysis for module 1 ##
#extractGOseq(test_annotation,1)
```

runSpinglass *Run Spinglass algorithm on a Scored PvalueAnnotation*

Description

This function is a function to prepare the data for calling the Spinglass network algorithm.

Usage

```
runSpinglass(pvalue_annotation, network, random_alpha = 0.05, gam = 0.5,
node_alpha = 0.05, maxsize = 500, minsize = 8, num_iterations = 1000, simplify = TRUE)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation
network	An graph object of class graphNEL or igraph
random_alpha	A numeric specifying a threshold with with to determine module signficance after randomization
gam	A parameter used by the Spinglass algorithm
node_alpha	The proportion of nodes to be used as seeds for the community detection
maxsize	The maximum module size
minsize	The minimum module size
num_iterations	The number of randomizations that will be computed to determine whether the module is significant by chance
simplify	A logical (TRUE(DEFAULT)/FALSE) that specifies whether network should be simplified by removing self loops and repeated edges

Details

In the provided Epimods reference, West et al outlined the advantages of using the spin-glass algorithm in the detection of modules. Please consult the reference for more detailed information on the spin-glass algorithm implemented in the package igraph.

Like Epimods, this function employs the spin-glass algorithm implemented in igraph and uses random permutations to assess the "modularity," the number and strength of connected nodes, of a module. However, SMITE scores are interpreted as Chi-square distributed statistics whenever possible, rather than the weighted-T-statistic in Epimods.

Value

An S4 object of class PvalueAnnotation with modules loaded

Note

This function was adapted from a function in the Epimods package that employs the spin-glass algorithm and uses random permutations to assess the "modularity" of a module . The original function was created by West et al.

Author(s)

N. Ari Wijetunga

References

James West, Stephan Beck, Xiangdong Wang & Andrew E. Teschendorff An integrative network algorithm identifies age-associated differential methylation interactome hotspots targeting stem-cell differentiation pathway. Scientific Reports 3, Article number: 1630 (2013)

https://code.google.com/p/epimods/

See Also

FEM runBioNet extractModules plotModule

Examples

```
data(test_annotation_score_data)

#load(system.file("data","Reactome.Symbol.Igraph.rda", package="SMITE"))

## NOTE: commented out for example.  See vignette for better explanation ##
#test_annotation <- runSpinglass(pvalue_annotation=test_annotation,
#network=REACTOME, maxsize=50, num_iterations=10)

plotModule(test_annotation, which_network=6, layout="fr")
```

---

scorePval *Making a single combined score for each gene*

Description

This function uses an a priori weighting scheme to combine scores for a given gene.

Usage

```
scorePval(pvalue_annotation, weights)
```

Arguments

pvalue_annotation	An S4 object of class PvalueAnnotation, for which makePvalueObject has already been run.
weights	A numeric vector of the relative importance of expression, modifications, and genomic contexts toward the final score. Names should be provided that match the "modfication_genomicfeature" format, except for expression. While the scores do not have to add up to 1, it is good practice to impose this restriction in order to track the relative importance.

Details

Because each weighting scheme generates scores from a distribution that will change depending on the analysis inputs, the function will randomly sample the final scores and compare each derived score to this simulated distribution.

If no names are given, then the function will assume the weights are in the order that it finds a particular "modification_genomicfeature" and it will print the weighting scheme so that you can verify it is correct. The total number of weights must match the total number of modifications*genomicfeatures+1 for expression.

After calculating a combined score (using a Stouffer's weighted statistic), a new p-value is derived using a non-parametric sampling approach.

Value

An S4 object of class PvalueAnnotation.

Author(s)

N. Ari Wijetunga

Examples

```
options(stringsAsFactors=FALSE)

data(test_annotation_score_data)

## NOTE: commented out for example.  See vignette for better explanation ##
#test_annotation<-scorePval(pvalue_annotation=test_annotation,
#weights=c(methylation_promoter=.3,methylation_body=.1,expression=.3,
#methylation_h3k4me1=.3))
```

stoufferTest *Stouffer's Test*

Description

This function performs a weighted Stouffer's mehtod of combining p-values.

Usage

```
stoufferTest(pvalues, weights)
```

Arguments

pvalues	A vector of p-values.
weights	Optional weights used when combining probabilites. If no weights are given then the p-values are equally weighted.

Details

For each p-value the inverse standard normal CDF is applied and Z scores are derived. Z-scores are then summed and a new Z score is transformed back to a p-value.

Value

A numeric p-value that represents the standard normal CDF of the combined Z statistic.

Note

This function was adapted from the function written on the Fisher's Method wikipedia page.

References

https://en.wikipedia.org/wiki/Fisher's_method

Stouffer S, DeVinney L, Suchmen E. The American soldier: Adjustment during army life. Vol. 1. Princeton University Press; Princeton, US: 1949.

Examples

```
## Generate test weights ##
weights<-runif(10, 1,100)
weights<-sort(weights)

## Generate test p-values##
pvals<-runif(10,0,1)

## run stoufferTest ##
stoufferTest(pvalues = pvals, weights=1/weights)
```

test_annotation_score_data *A toy PvalueAnnotation*

Description

This Pvalue annotation has only scoring data filled in to use in late pipeline "SMITE" functions. It can be used to skip the loading data phase of analysis and test latter functionality.

Usage

```
data("test_annotation_score_data")
```

Format

A PvalueAnnotation with the following slots

score_data a PvalueObject with slots corresponding to pval_data, effect_data, genes, signs_index, scores, trans, scoring_vector, and module_otuput

Details

This is a PvalueAnnotation which has had all of the pre-scoring data removed so that it is only usefule for using functions beginning with SMITE and SMITE plotting functions.

Value

A PvalueAnnotation with the score_data slot containing toy scores

Examples

```
data(test_annotation_score_data)
plotDensityPval(test_annotation)
head(extractScores(test_annotation))
```

Index