Package 'rtfbsdb'

June 8, 2015

Version 0.1.5

Date 2015-04-27

Title Parse TF motifs from public databases, read into R, and scan using 'rtfbs'.

Author Charles G. Danko `<dankoc@gmail.com>`, Zhong Wang`<zw355@cornell.edu>`

Maintainer Charles G. Danko `<dankoc@gmail.com>` Zhong Wang`<zw355@cornell.edu>`

Depends R (>= 2.6)

Imports rphast, rtfbs, bigWig, parallel, grid, cluster, methods, latticeExtra,lattice, tools

LinkingTo

Suggests RCurl, stringr

Description
Convenience functions to read and scan DNA sequences using Position Weight Matrices (PWMs)

License GPL version 3 or newer

biocViews Sequencing, Analysis

LazyLoad yes

R topics documented:

`CisBP.db-class` *Class* `"CisBP.db"`

Description

The motif library from CisBP web site.
Link: `http://cisbp.ccbr.utoronto.ca/`

Objects from the Class

Objects can be created by calls of the form `CisBP.extdata`, `CisBP.zipload`, `CisBP.download`.

Slots

`species`: String indicating the species name defined in the CisBP dataset.

`zip.file`: String indicating the filename of temporary data file.

`zip.url`: String indicating the download source.

`file.tfinfo`: String indicating the TF filename, defulat is TF_Information.txt.

Extends

Class `"tfbs.db"`, directly.

Methods

tfbs.createFromCisBP Build a tfbs object by querying the meta file of CisBP dataset and subsetting the results.

CisBP.group Get the statistical summary by grouping the field in the CisBP dataset.

References

Weirauch, M. T., Yang, A., Albu, M., Cote, A. G., Montenegro-Montero, A., Drewe, P., ... & Hughes, T. R. (2014). Determination and inference of eukaryotic transcription factor sequence specificity. Cell, 158(6), 1431-1443.

Examples

```
showClass("CisBP.db")
```

`CisBP.download` *Download CisBP dataset.*

Description

Download TF data file from CisBP dataset and store it to temporary folder

Usage

```
CisBP.download(species = "Homo_sapiens",
      url = "http://cisbp.ccbr.utoronto.ca/bulk_archive.php")
```

Arguments

`species`	String, indicating the species name in the CisBP dataset
`url`	String, the URL of bulk dowbnloads from CisBP dataset, default is `http://cisbp.ccbr.utoronto.ca/bulk_archive.php`

Value

A CisBP object (class name: "`CisBP.db`") is returned with four items:

`species`	String indicating the species name
`zip.file`	String indicating the filename of temporary data file.
`zip.url`	String indicating the download source
`file.tfinfo`	String indicating the TF filename, default is TF_Information.txt.

References

Weirauch, M. T., Yang, A., Albu, M., Cote, A. G., Montenegro-Montero, A., Drewe, P., ... & Hughes, T. R. (2014). Determination and inference of eukaryotic transcription factor sequence specificity. Cell, 158(6), 1431-1443.

See Also

See Also as `CisBP.zipload`, `CisBP.extdata`.

Examples

```

#download human dataset
db1 <- CisBP.download("Homo_sapiens");

#download mouse dataset
db2 <- CisBP.download("Mus_musculus");
```

`CisBP.extdata` *Load internal CisBP dataset.*

Description

Build a CisBP object from the internal zip file stored in this package

Usage

```
CisBP.extdata(species)
```

Arguments

`species`	String, only valid for human and mouse species, i.e. Homo_sapiens or Mus_musculus

Value

A CisBP object (class name: `"CisBP.db"`) is returned with four items:

`species`	String indicating the species name defined in the CisBP dataset.
`zip.file`	String indicating the filename of temporary data file.
`zip.url`	String indicating the download source
`file.tfinfo`	String indicating the TF filename, default is TF_Information.txt.

See Also

See Also as `CisBP.zipload`, `CisBP.download`.

Examples

```
#reading data from inner file
db.human <- CisBP.extdata("Homo_sapiens")
```

`CisBP.group` *Summarize the motif number.*

Description

Get the statistical summary by grouping the field in the CisBP dataset.

Usage

```
CisBP.group(cisbp.db,
      group.by=c("tf_name", "tf_species", "tf_status", "family_name",
            "motif_type", "msource_id"),
      tf.information.type=1)
```

Arguments

`cisbp.db`	A CisBP object (`"CisBP.db"`) including the TF_Information.txt.
`group.by`	String, indicating which field will be used to group values. Available values are tf_name, tf_species, tf_status, family_name, motif_type and msource_id.
`tf.information.type`	Number, indicating which TF meta file will be used. Available values are 1 for TF_Information.txt, 2 for TF_Information_all_motifs.txt and 3 for F_Information_all_motifs_plus.txt.

Details

Three TF information files in CisBP dataset.

1: TF_Information.txt : (direct motifs) or (no direct but inferred motifs with 90%)
2: TF_Information_all_motifs.txt: (direct motifs) and (inferred motifs above the threshold)
3: F_Information_all_motifs_plus.txt: All motifs

Value

A data frame returned includes two columns

`group_by`	Values of grouping field
`number`	Counts of group value

See Also

See Also as `tfbs.createFromCisBP`

Examples

```
# Load the internal CisBP dataset
db_human <- CisBP.extdata("Homo_sapiens");

# Group the motif count by the column of family_name in TF_Information.txt
gr1 <- CisBP.group(db_human, group.by="family_name", tf.information.type=1 );

# Group the motif count by the column of tf_status in TF_Information.txt
gr2 <- CisBP.group(db_human, group.by="tf_status", tf.information.type=1 );

# Group the motif count by the column of tf_status in TF_Information_all_motifs.txt
gr3 <- CisBP.group(db_human, group.by="tf_status", tf.information.type=2);

# Group the motif count by the column of tf_status in F_Information_all_motifs_plus.txt
gr4 <- CisBP.group(db_human, group.by="tf_status", tf.information.type=3);
```

`CisBP.zipload` *Load the zipped CisBP file.*

Description

Build a CisBP object from the zipped CisBP file.

Usage

```
CisBP.zipload(zip.file, species = "Homo_sapiens")
```

Arguments

`zip.file`	String, indicating the zipped file data
`species`	String, indicating the species name in the CisBP database

Value

A CisBP object (class name: `"CisBP.db"`) is returned with four items:

`species`	String indicating the species name
`zip.file`	String indicating the filename of temporary data file.
`zip.url`	String indicating the download source
`file.tfinfo`	String indicating the TF filename, default is TF_Information.txt.

See Also

See Also as `CisBP.extdata`, `CisBP.download`.

Examples

```
# Download the dataset
db2 <- CisBP.download("Mus_musculus");

# Loading the zip file, the db2 and db3 have same TF data.
# Here is an example to show how to use CisBP.zipload.
# We dont nee to download it by CisBP.download and then load it by CisBP.zipload
db3 <- CisBP.zipload(db2@zip.file, species="Mus_musculus");
```

`tfbs`	*Create a tfbs object from the supplied PWM files.*

Description

Create a tfbs object from the supplied PWM files.

Usage

```
tfbs(filenames,
      names,
      species="Homo_sapiens",
      extra_info = NULL, ...)
```

Arguments

`filenames`	Vector of PWM files
`names`	Vector of unique gene symbols.
`species`	String indicating species name
`extra_info`	Data frame including meta information for all motifs., Default: NULL
`...`	Parameters,such as pseudocount, force_even, and the parameters used in read.table function.

Value

A tfbs object (class: "`tfbs`") including all PWM matrics.The all attributes are as follows:

`TFID`	Vector of non-unique ID for TF.
`species`	String indicating the species name
`ntfs`	Number of motifs in matrix.
`pwm`	A list including PWM matics.
`filename`	Vector of PWM filename.
`mgisymbols`	Unique gene symbols for TF.
`extra_info`	Data frame, including extra information for PWMs, it maybe different with motif dataset, default:NULL.
`distancematrix`	Distance matrix between motifs returned by `tfbs.getDistanceMatrix`, default:NULL.
`expressionlevel`	Data frame indicatig the result of expression level returned by `tfbs.getExpression`, default:NULL.

The tfbs object can be created by the function of `tfbs`, `tfbs.dir`, `tfbs.createFromCisBP`.

Examples

```
# M3590_1.01 PAX5 ENSG00000196092
# M3590_1.01 PAX5 ENSG00000196092
fs1 <- system.file("extdata","M3590_1.01.pwm", package="rtfbsdb")
fs2 <- system.file("extdata","M3591_1.01.pwm", package="rtfbsdb")

cat(fs1, "\n");

tfs <- tfbs( c( fs1, fs2 ), names=c("M3590_1.01","M3591_1.01"),
      header=TRUE, sep="\t" , row.names=1 );
str(tfs);
```

`tfbs-class` *Class* `"tfbs"`

Description

Tfbs object is a collection of motif PWM data. Some functions are provided based on the PWM and GENCODE data, such as clustering, search and compare.

Objects from the Class

Objects can be created by calls of the function of `tfbs.createFromCisBP`, `tfbs.dirs` and `tfbs`.

Slots

TFID Vector of non-unique ID for TF.

species String indicating the species name

ntfs Number of motifs in matrix.

pwm A list including PWM matics.

filename Vector of PWM filename.

mgisymbols Unique gene symbols for TF.

extra_info Data frame, including extra information for PWMs, it maybe different with motif dataset, default:NULL.

distancematrix Distance matrix between motifs returned by `tfbs.getDistanceMatrix`, default:NULL.

expressionlevel Data frame indicatig the result of expression level returned by `tfbs.getExpression`, default:NULL.

Methods

tfbs.getDistanceMatrix Calcuate a distance matrix with Pearson's R values

tfbs.getExpression Estimate gene expression of target TF.

tfbs.clusterMotifs Cluster the specified motifs and drawing the heatmap.

tfbs.scanTFsite Find TF sites from genome data within the BED ranges.

tfbs.compareTFsite Comparative TFBS search with the BED ranges

tfbs.selectByGeneExp Select the motifs with minimum p-value from each group of clustering.

tfbs.selectByRandom Select the motifs randomly from each group of clustering.

tfbs.drawLogosForClusters Draw the motif logos by one group per page.

tfbs.drawLogo Draw the logo for a single TF motif.

Examples

```
showClass("tfbs")
```

`tfbs.clusterMotifs` *Clustering the specified motifs and drawing the heatmap.*

Description

Clustering the specified motifs and drawing the heatmap.

Usage

```
tfbs.clusterMotifs(tfbs,
      subset = NA,
      pdf.heatmap = NA,
      method = NA,
      group.k = NA)
```

Arguments

`tfbs`	A tfbs object (`"tfbs"`) returned by `tfbs.createFromCisBP`, `tfbs.dirs` or other functions.
`subset`	Vector, the indexes of partial motifs if not all motifs are clustered.
`pdf.heatmap`	String, a PDF filename for heatmap.
`method`	String, availabe values are "agnes" and "cors".
`group.k`	Integer, if the method of agnes is used to do clustering, the parameter of k is optional to use as preset group number.

Details

This result of clustering will be used in the

Value

A matrix with 2 columns is returned, 1st column is the index of motifs and 2nd column is the group number of clustering.

See Also

See Also as `tfbs.selectByGeneExp` and `tfbs.selectByRandom`

Examples

```
# Load the internal CisBP data set
db <- CisBP.extdata("Homo_sapiens");

# Create a tfbs object by querying the meta file of CisBP dataset.
tfs <- tfbs.createFromCisBP(db, motif_type="ChIP-seq", tf.information.type=1 );

# Calculate the distance matrix
tfs <- tfbs.getDistanceMatrix( tfs, ncores=1 );

# Cluster the motifs using the "cors" method
cluster1 <- tfbs.clusterMotifs(tfs, pdf.heatmap = "test-heatmap1.pdf", method="cors" );
show(cluster1);

# draw motif logos on one group per page.
tfbs.drawLogosForClusters(tfs, cluster1, "test-cluster1.pdf")

# Cluster the motifs using the "agnes" function
cluster2 <- tfbs.clusterMotifs(tfs, pdf.heatmap = "test-heatmap2.pdf", method="agnes" );
show(cluster2);
```

`tfbs.compareTFsite` *Comparative TFBS search with the BED ranges*

Usage

```
tfbs.compareTFsite(tfbs,
       file.twoBit,
       positive.bed,
       negative.bed,
       file.prefix,
       usemotifs=NA,
       background.correction=FALSE,
       fdr=0.1,
       threshold=NA,
       background.order=2,
       background.length=100000,
       ncores=3)
```

Arguments

`tfbs`	A tfbs object, see also "`tfbs`"
`file.twoBit`	String, the file name of genome data(hg19.2bit, mm10.2bit)
`positive.bed`	Data frame, bed-formatted down-regulatory ranges
`negative.bed`	Data frame, bed-formatted up-regulatory ranges
`file.prefix`	String, the prefix for outputted file
`usemotifs`	Number, the index of which motifs are used to compare. All motifs are used if NA.

`background.correction`	Logical value, if the difference between positive and negative TREs is significant,the resampling will be applied to the correction for the negative TREs. Default:FALSE.
`fdr`	Numeric value, False Discovery Rate (FDR) of possible binding sites, only binding sites with FDR less than this value can be selected.
`threshold`	Numeric value, only sites with scores above this threshold are returned (default = 6)
`background.order`	Number, order of Markov model to build background
`background.length`	Number, length of the sequence to simulate background
`ncores`	Number, comupting nodes in parallel environment.

Details

The background difference between positive.bed and negative.bed is checked before the comparson. The p-value of Wilcox test and a vioplot figure show this difference and help the user to determine whether the correction is necessary. If the difference is very significant, please set *background.correction* to do background correction by resampling the TREs from negative bed data based on the frequency of TREs in negative bed data.

Value

A data frame with the following columns:

`tf.name`	TF name.
`Npos`	TF site count found in positive ranges.
`Nneg`	TF site count found in negative ranges.
`es.ratio`	Enrichment score.
`assoc.pvalue`	p-value calculated by fisher test.
`pv.bonferoni`	p-value corrected by Bonferoni.
`starch`	Binary filename of detected TF sites.
`motif.id`	Motif ID.

Examples

```
file.dREG.H.change.bed <- "/home/zw355/src/rtfbs_db/rtfbsdb/test/dREG.H.change.bed"
file.dREG.all.bed    <- "/home/zw355/src/rtfbs_db/rtfbsdb/test/dREG.all.bed"
file.twoBit     <- "/local/storage/data/hg19/hg19.2bit"

db <- CisBP.extdata("Homo_sapiens");
tfs <- tfbs.createFromCisBP(db, family_name="AP-2");

dREG_H_change_bed <- read.table(file.dREG.H.change.bed, header=FALSE);
dREG_all_bed <- read.table(file.dREG.all.bed, header=FALSE);

t <- tfbs.compareTFsite( tfs,
      file.twoBit,
      dREG_H_change_bed,
      dREG_all_bed,
      background.correction=TRUE,
      file.prefix="comp.db",
      ncores = 1); #ncores=3
```

`tfbs.createFromCisBP` *Create TF object by querying the CisBP dataset.*

Description

Build a tfbs object by querying the meta file of CisBP dataset and subsetting the results.

Usage

```
tfbs.createFromCisBP(cisbp.db,
      tf_name = NULL,
      tf_status = NULL,
      family_name = NULL,
      motif_type = NULL,
      msource_id = NULL,
      tf.information.type = 1,
      expressed.only=TRUE,
      include.DBID.Missing=TRUE,
      seq.datatype=NA,
      file.bigwig.plus=NA,
      file.bigwig.minus=NA,
      file.twoBit=NA,
      file.gencode.gtf=NA,
      ncores = 1 )
```

Arguments

`cisbp.db`	A CisBP object("`CisBP.db`"), including the file of TF_Information.txt.
`tf_name`	String, indicating the TF_name field will be used to select motifs.
`tf_status`	String, indicating the TF_Status field will be used to select motifs.
`family_name`	String, indicating the Family_Name field will be used to select motifs.
`motif_type`	String, indicating the Motif_Type field will be used to select motifs.
`msource_id`	String, indicating the MSource_Identifier field will be used to select motifs.
`tf.information.type`	Number,indicating which TF meta file will be used. Available values are 1 for TF_Information.txt, 2 for TF_Information_all_motifs.txt and 3 for F_Information_all_motifs_plus.txt.
`expressed.only`	Logical, indicating the only expressed TFs are selected to construct this object based on the gene exprssion values.
`include.DBID.Missing`	Logical, indicating whether the TFs without association with GENCODE through the DBID are selected.

`seq.datatype`	String,indicating which kind of seq data is applied to this function, three values are available: GRO-seq, PRO-seq and RNA-seq. Default: GRO-seq
`file.bigwig.plus`	String, indicating bigwig file for strand plus(+).
`file.bigwig.minus`	String, indicating bigwig file for strand minus(-).
`file.twoBit`	String, indicating the binary data of sequence.
`file.gencode.gtf`	String,indicating Gencode GTF file downloaded from the Gencode web site.
`ncores`	Number, comuputing nodes in parallel environment for gencode data converting.

Details

The function includes three steps to build a tfbs object:
1) Searching the TF information and PWM files in the CisBP dataset according to the criteria specified by the parameters of *tf_name*, *tf_status*, *family_name*, *motif_type* and *msource_id*.
2) If the bigwig files are provided, the gene expression values are calculated through querying the TREs region from the GENCODE database(for human, gencode.v21.annotation.gtf, for mouse: gencode.vM3.annotation.gtf) and querying the reads count in the plus and minus bigWig files.
3) If the expressed TFs only is used in the tfbs object, the TFs with p-values less than 0.05 will be selected.

The following part explains how to calculate the gene expression.

For each motif, the occurance ranges can be queried by the gene ID After the searching, one range obtained from the merge of the multiple ranges will be used to detect the reads count in the specified bigwig files(including plus and minus). The probability of each motif can be calcuated by the reads count and lambda.

The lambda is determined by the following formulation:

```
r.lambda = 0.04 * sum(reads_in_all_chromosomes)/10751533/1000.
```

The dataset of GENECODE v21 (human) and vM3 (mouse) have been compiled into RDATA file and attached in this package.

The `gencode_transcript_ext` object can be accessed after the following command is executed successfully.

```
load( system.file("extdata", "gencode_v21_transcript_ext.rdata",
package="rtfbsdb"), environment() );
```

Value

A tfbs object is returned with PWM matrices, see Also as "`tfbs`"

See Also

See Also as `tfbs`

Examples

```
# Load the internal CisBP dataset
db_human <- CisBP.extdata("Homo_sapiens");

# Load all motifs and return a tfbs object.
tfs0 <- tfbs.createFromCisBP(db_human);

# Query the motifs by the conditions and return a tfbs object
tfs1 <- tfbs.createFromCisBP(db_human, family_name="Homeodomain", tf_status="D",
      motif_type="ChIP-seq", msource_id= "MS01_1.01", tf.information.type=1 );

# Query the motifs by the conditions and return a tfbs object
tfs2 <- tfbs.createFromCisBP(db_human, family_name="Homeodomain", tf_status="D" );

# Query the motifs by the conditions and return a tfbs object
tfs3 <- tfbs.createFromCisBP(db_human, motif_type="ChIP-seq"  );

# Query the motifs by the conditions and return a tfbs object
tfs4 <- tfbs.createFromCisBP(db_human, tf.information.type=2);
```

`tfbs.db-class` *Class* `"tfbs.db"`

Description

Abstract class for motif dataset. The CisBP class is a son class of tfbs.db.

Objects from the Class

Now code or function can be used to create this class.

Slots

`species`: Species name.

Methods

No methods defined with class "tfbs.db" in the signature.

See Also

`"CisBP.db"` inherits this class.

Examples

```
showClass("tfbs.db")
```

`tfbs.dirs` *Create a tfbs object from the folders.*

Description

Create a tfbs object from all the PWM files found in the supplied folders.

Usage

```
tfbs.dirs(...,
      species = "Homo_sapiens",
      args.read.motif = NULL,
      pattern = glob2rx("*.pwm"),
      recursive = FALSE)
```

Arguments

`...`	Multiple strings, one or more folders can be used in this function.
`species`	String, including the species name.
`args.read.motif`	List, including *pseudocount*, *force_even* or other parameters used in `read.table` function.
`pattern`	String, a character vector specifying regular expression and wlidcards.
`recursive`	Logical, indicating the loading recursively descends into subfolders or not, default: FALSE.

Details

Two parameters in the list of `args.read.motif` can be used:
pseudocount: log value for zero value in PWM matrix, default is -7.
force_even: whether the PWM matrix with odd size needs to be even.

Value

A tfbs object collecting all the PWM files in the specified folders. For the details of tfbs object, please see `tfbs`

See Also

The structure of tfbs object is described in "`tfbs`"

Examples

```
fs.dir <- system.file("extdata","", package="rtfbsdb")
tfs <- tfbs.dirs( fs.dir,
      args.read.motif = list(pseudocount=-7, header=TRUE, sep="\t" , row.names=1) );
str(tfs);
```

tfbs.drawLogo *Draw single motif logo*

Description

Draw the logo for a single TF motif.

Usage

```
tfbs.drawLogo(tfbs, index)
```

Arguments

`tfbs`	A tfbs object("`tfbs`")
`index`	Vector of number, indicating the motif index.

Value

No return values.

See Also

See Also as "`tfbs`"

Examples

```
db <- CisBP.extdata("Homo_sapiens");
tfs <- tfbs.createFromCisBP(db, family_name="AP-2");
pdf("test-logos.pdf");
tfbs.drawLogo(tfs, 1:10 );
dev.off();

#unlink("test-logos.pdf");
```

tfbs.drawLogosForClusters *Draw the motif logos by clustering.*

Description

Draw the motif logos by one group per page.

Usage

```
tfbs.drawLogosForClusters(tfbs, cluster.mat, pdf.logos)
```

Arguments

`tfbs`	A tfbs object("`tfbs`").
`cluster.mat`	A matrix with 2 columns returned by `tfbs.clusterMotifs`, 1st column is the index of motifs and 2nd column is the group number of clustering.
`pdf.logos`	String indicating a PDF eilname.

Value

No return value.

See Also

See Also as `tfbs.clusterMotifs`

Examples

```
# Load the internal CisBP data set
db <- CisBP.extdata("Homo_sapiens");

# Create a tfbs object by querying the meta file of CisBP dataset.
tfs <- tfbs.createFromCisBP(db, motif_type="ChIP-seq", tf.information.type=1 );

# Calculate the distance matrix
tfs <- tfbs.getDistanceMatrix( tfs, ncores=1);

# Cluster the motifs using the "cors" method
cluster1 <- tfbs.clusterMotifs(tfs, pdf.heatmap = "test-heatmap1.pdf", method="cors" );
show(cluster1);

# draw motif logos on one group per page.
tfbs.drawLogosForClusters(tfs, cluster1, "test-cluster1.pdf")
```

`tfbs.getDistanceMatrix` *Calcuate a distance matrix with Pearson's R values.*

Description

Compare any two motifs and return a matrix with Pearson's R values.

Usage

```
tfbs.getDistanceMatrix(tfbs, ncores = 3, BG = log(c(0.25, 0.25, 0.25, 0.25)))
```

Arguments

`tfbs`	A tfbs object("`tfbs`").
`ncores`	Number, the number of cores to use simultaneously.
`BG`	The log value of probabilities for nucleotide A, C, G and T as Backgroud computing.

Value

A tfbs object with new distance matrix (@distancematrix).

Examples

```
db <- CisBP.extdata("Homo_sapiens");
tfs <- tfbs.createFromCisBP(db, family_name="AP-2");
tfs0 <- tfbs.getDistanceMatrix(tfs, ncores=1);
```

`tfbs.getExpression` *Estimate gene expression of target TF.*

Description

Gets expression level of target TF.
USE extra_info$DBID to find gene information encoded by GENCODE V21

Usage

```
tfbs.getExpression(tfbs,
      file.bigwig.plus, file.bigwig.minus,
      file.twoBit=NA,
      file.gencode.gtf=NA,
      seq.datatype=NA,
      ncores =1 )
```

Arguments

`tfbs`	A tfbs object("`tfbs`").
`file.bigwig.plus`	String, indicating bigwig file for strand plus(+).
`file.bigwig.minus`	String, indicating bigwig file for strand minus(-).
`file.twoBit`	String, indicating the binary data of sequence.
`file.gencode.gtf`	Gencode RDATA file encoded by ths package.
`seq.datatype`	String,indicating which kind of seq data is applied to this function, three values are available: GRO-seq, PRO-seq and RNA-seq. Default: GRO-seq
`ncores`	Number, comupting nodes in parallel environment.

Details

For each motif, the occurance ranges can be queried by the gene ID in the GENCODE database(for human, gencode.v21.annotation.gtf, for mouse: gencode.vM3.annotation.gtf). After the searching, one range obtianed from the merge of the multiple ranges will be used to detect the reads count in the specified bigwig files(including plus and minus). The probability of each motif can be calcuated by the reads count and lambda.

The lambda is determined by the following formulation:

```
r.lambda = 0.04 * sum(reads_in_all_chromosomes)/10751533/1000.
```

The dataset of GENECODE v21 (human) and vM3 (mouse) have been compiled into RDATA file and attached in this package.

The `gencode_transcript_ext` object can be accessed after the following command is executed successfully.

```
load( system.file("extdata", "gencode_human21_transcript_ext.rdata",
package="rtfbsdb"), environment() );
```

Value

A tbfs object with new expression data frame including the follwing columns:

`Motif_ID`	Motif_ID from CisBP dataset or other data source.
`DBID`	DBID from CisBP dataset or other data source.
`chr`	String, chromosome name.
`start`	Integer, start postion in which gene ID can be detected.
`end`	Integer, end postion in which gene ID can be detected.
`strand`	String, + or -, indicating the strand direction.
`bed_length`	Integer, the length of range which gene ID can be detected.
`reads`	The reads number queried by BigWig function from the bigwig files(plus and minus)
`lambda`	The lambda parameter in poison distribution.
`prob`	The probability calculated based on Poisson distribution.

See Also

See Also as "`tfbs`", ~~~

Examples

```
# Load the internal CisBP data set
db.human <- CisBP.extdata("Homo_sapiens");

# Create a tfbs object by querying the meta file of CisBP dataset.
tfs <- tfbs.createFromCisBP(db.human, motif_type="ChIP-seq", tf.information.type=1 );
```

```
if(0)
{
file.bigwig.plus <- "testdata//GSM1480327_K562_PROseq_plus.bw";
file.bigwig.minus <-"testdata/GSM1480327_K562_PROseq_minus.bw";

tfs <- tfbs.getExpression(tfs, file.bigwig.plus, file.bigwig.minus,
       file.gencode.gtf="/local/storage/data/gencode/gencode.v21.annotation.gtf" );

tfs <- tfbs.getExpression(tfs, file.bigwig.plus, file.bigwig.minus, ncores = 21 );
}
```

`tfbs.scanTFsite` *Find TF sites from genome data within the BED ranges*

Description

Find TF sites from genome data within the BED ranges. Please notice that this package does not provided genome data such as hg19.2bit, mm10.2bit.

Usage

```
tfbs.scanTFsite(tfbs,
       file.twoBit,
       dnase.peaks.bed=NULL,
       file.prefix=NA,
       usemotifs=NA,
       ncores=3,
       return.type=c("matches", "posteriors", "maxposterior", "writedb"),
       threshold=6, ...)
```

Arguments

`tfbs`	A tfbs object (`"tfbs"`) returned by `tfbs.createFromCisBP`, `tfbs`, `tfbs.dirs`.
`file.twoBit`	String, the file name of genome data(hg19.2bit or mm10.2bit)
`dnase.peaks.bed`	Data frame, bed-formatted peak information
`file.prefix`	String, the prefix for outputted file, only used when the return.type is *writedb*
`usemotifs`	Vector indicating indexes of motif to be used in scanning.
`ncores`	Number, computing nodes in parallel environment.
`return.type`	String, four available values explained in th details
`threshold`	Numeric value, only sites with scores above this threshold are returned (default = 6)
`...`	Any parameters used in the function of score.ms.

Details

(1) Four options are availabe for the function of tfbs.scanTFsite as follows.

- matches: returns all matching motifs.
- writedb: writes a bed file with matches. Assumes that sort-bed and starch tools are availiable in $PATH
- posteriors: returns the posteriors at each position.
- maxposterior: returns the max(posterior) in each dnase-1 peak.

(2) In order to make the binary file with the parameter of writedb, make sure that starchcat and sort-bed command (in BEDOPS) can be accessed from R environment. If not, please put the folder in $PATH.

Value

The option of *matches* returns a list including the result of every motif, which result is BED style data frame with the following columns.

`chrom`	chromosome
`chromStart`	start position
`chromEnd`	chromosome end position
`name`	
`score`	
`strand`	strand

The option of *writedb* will return a binary BED filename in which store all bed ranges.

The option of *posteriors* will return a list for each motif returned by score.ms function. Scores represent the motif 'match score', or the product of the probability of observing each base under the motif or background models. Scores are returned under the motif model for all positions in the sequence, on both forward and reverse strands, and under the background model.

The option of *maxposterior* will return a probability matrix which the row indicates the range of dnase peak and the column indicates the motif.

Examples

```
library(rtfbsdb);

file.dREG.H.change.bed <- "/home/zw355/src/rtfbs_db/rtfbsdb/test/dREG.H.change.bed"
file.twoBit      <- "/local/storage/data/hg19/hg19.2bit"

db <- CisBP.extdata("Homo_sapiens");
tfs <- tfbs.createFromCisBP(db, family_name="AP-2");

dREG_H_change_bed <- read.table(file.dREG.H.change.bed, header=FALSE);
```

```
t0 <- tfbs.scanTFsite( tfs,
file.twoBit,
dREG_H_change_bed,
file.prefix="test.db",
ncores = 1);

str(t0[[1]])

t1 <- tfbs.scanTFsite( tfs,
file.twoBit,
dREG_H_change_bed,
file.prefix="test.db",
return.type="writedb",
ncores = 1);

t1

t2 <- tfbs.scanTFsite( tfs,
file.twoBit,
dREG_H_change_bed,
file.prefix="test.db",
return.type="posteriors",
ncores = 1);

str(t2[[1]]);

t3 <- tfbs.scanTFsite( tfs,
file.twoBit,
dREG_H_change_bed,
file.prefix="test.db",
return.type="maxposterior",
ncores = 1);

str(t3);
```

`tfbs.selectByGeneExp` *Motif selection by gene expression level.*

Description

Select the motifs with minimum p-value from each group of clustering.

Usage

```
tfbs.selectByGeneExp(tfbs, cluster.mat)
```

Arguments

`tfbs`	A tfbs object ("`tfbs`") with the data frame of gene expression level.
`cluster.mat`	A matrix with 2 columns returned by `tfbs.clusterMotifs`, 1st column is the index of motifs and 2nd column is the group number of clustering.

Details

The function of `tfbs.getExpression` should be successfully called and the results of gene expression should be returned before this function is called. The indexes of selected motifs will be used in the function of `tfbs.compareTFsite` or `tfbs.scanTFsite`.

Value

A vector of motif indices is returned.

See Also

See Also as `tfbs.selectByRandom`, `tfbs.getExpression`

Examples

```
db <- CisBP.extdata("Homo_sapiens");

tfs <- tfbs.createFromCisBP(db, family_name="AP-2");

if(0)
{
tfs <- tfbs.getExpression(tfs, file.bigwig.plus, file.bigwig.minus, file.hg19);

cluster1 <- tfbs.clusterMotifs(tfs, pdf.heatmap="test-AP2-heatmap.pdf" );

usemotif <- tfbs.selectByGeneExp(tf, cluster1);
}
```

`tfbs.selectByRandom` *Random motif selection*

Description

Select the motifs randomly from each group of clustering.

Usage

```
tfbs.selectByRandom(tfbs, cluster.mat)
```

Arguments

`tfbs`	A tfbs object("`tfbs`").
`cluster.mat`	A matrix with 2 columns returned by `tfbs.clusterMotifs`, 1st column is the index of motifs and 2nd column is the group number of clustering.

Details

The indexes of selected motifs can be used in the function of `tfbs.compareTFsite` or `tfbs.scanTFsite`.

Value

A vector of motif indices is returned.

See Also

See Also as `tfbs.selectByGeneExp`, `tfbs.getExpression`

Examples

```
db <- CisBP.extdata("Homo_sapiens");

tfs <- tfbs.createFromCisBP(db, family_name="AP-2");

tfs <- tfbs.getDistanceMatrix(tfs, ncores=1);

cluster1 <- tfbs.clusterMotifs(tfs, pdf.heatmap="test-AP2-heatmap.pdf" );

usemotif <- tfbs.selectByRandom(tfs, cluster1);

show(usemotif);
```

Index