

MotIV

October 5, 2010

`alignments-class` *Class "alignments"*

Description

This object contains the alignments found by a MotIV analysis.

Objects from the Class

Objects can be created by calls of the form `new("alignments", TF, evalue, sequence, match, strand)`.

Slots

TF Object of class "TF"

evalue The e-value of the alignment.

sequence The input sequence aligned.

match The TF sequence which has been matched.

strand The strand of the alignment.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

`motiv`, `matches`, `transcriptionFactor`

combine

*Combine Motifs***Description**

This function combines motifs according to a set of filters.

Usage

```
## S4 method for signature 'motiv,filters':
combine(x, f, name=NULL, exact=FALSE, verbose=TRUE)
```

Arguments

x	An object of class <code>motiv</code> .
f	A filter or a set of filter.
name	Name(s) to be given for similar motifs.
verbose	If <code>FALSE</code> , no output will be print.
exact	If <code>TRUE</code> , search only for perfect name match.

Details

This function is used to consider some motifs as a unique motif or `similar motifs`.

Many `filters` could be pass in argument separated by coma. They will be considered independently (coma is considered as OR).

If a name or a vector of name is provided, it will be used to assign new name for similar motif to the corresponding filter. Else, a generic name is used.

Value

A `motiv` object.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

`setFilter`, `filter`, `split`

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)
```

```
#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOX
```

exportAsRangedData *Export MotIV Results*

Description

Export your

Usage

```
exportAsRangedData(x, y, bysim=TRUE, correction=TRUE)
```

Arguments

x	An object of class <code>motiv</code> .
y	The <code>rGADEM</code> type object associated with the <code>motiv</code> object.
bysim	If <code>TRUE</code> , the 'similar' field (defined with the <code>combine</code> function) will be print instead of the original name.
correction	If <code>TRUE</code> , corrects the position according to the alignment.

Details

Use this function to export the results into a `RangedData` object.

Value

An object of type `RangedData`.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
```

```

motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.split <- split(foxa1.analysis.jaspar, c(f.foxa1, f.ap1) , drop=FALSE, exact=FALSE,
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOX

#####Plots#####
#plot(foxa1.filter.combine, ncol=2,top=5, rev=FALSE, main="FOXA1", bysim=TRUE)
#plot(foxa1.filter.combine ,gadem,ncol=2, type="distribution", correction=TRUE, group=FA
#plot(foxa1.filter.combine ,gadem,type="distance", correction=TRUE, group=TRUE, bysim=TRU

#####RangedData#####
foxa1.rd <- exportAsRangedData(foxa1.filter.combine["FOXA1"], gadem)
ap1.rd <- exportAsRangedData(foxa1.filter.combine["AP1"], gadem)

```

exportAsTransfacFile

Write Transfac Files

Description

Export an object of class `motiv` as a Transfac file type.

Usage

```
exportAsTransfacFile(x, file)
```

Arguments

<code>x</code>	An object of class <code>motiv</code> to be export.
<code>file</code>	A character string naming a file.

Details

This function is made to provided standard output file used by STAMP. It take an object of class `motiv` and write two files named `*_matched.txt` and `*_match_pairs.txt` containing alignments and identified PWMs.

For more information about the Transfac file format, please refere to <http://www.benoslab.pitt.edu/stamp/help.html>.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.split <- split(foxa1.analysis.jaspar, c(f.foxa1, f.ap1) , drop=FALSE, exact=FALSE,
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOX

#####Export#####
#exportAsTransfacFile(foxa1.filter.combine, file="foxa1_analysis")
```

filter-class	<i>Class "filter"</i>
--------------	-----------------------

Description

This object information to be apply as filter.

Details

This class `filter` is used to selected `motif` objects according the filter's arguments.

Objects from the Class

Objects can be created by calls of the form `new("filter", name, tfname, top, evaluateMax, lengthMax, valid)`.

Slots

- name** A name or a list of names.
- tfname** A transcription factor name or a list of TF names.
- evaluateMax** An e-value between 0 and 1.
- top** Defined the depth of the filter.
- lengthMax** The maximum motif length.
- valid** The alignment that should be considered as valid.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

setFilter, filter, split, combine

Examples

```
showClass("filter")
```

filters-methods *Filters Methods*

Description

Methods for filters object

Usage

```
## S4 method for signature 'filter':  
summary(object)  
## S4 method for signature 'filters':  
summary(object)
```

Arguments

object An object of class `motiv`.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

setFilter, filter, split, combine

Examples

```
showClass("filter")
```

`filter`*Filter Motifs*

Description

This function selects motifs according to a set of filters.

Usage

```
## S4 method for signature 'motiv, filters':  
filter(x, f, exact=FALSE, verbose=TRUE)
```

Arguments

<code>x</code>	An object of class <code>motiv</code> .
<code>f</code>	A filter or a set of filter for <code>motiv</code> object.
<code>verbose</code>	If <code>FALSE</code> , no output will be print.
<code>exact</code>	If <code>TRUE</code> , search only for perfect name match.

Details

This function is used to select motifs that correspond to the filters.

Many `filter` could be pass in argument separated by coma. They will be considered independently.

Value

A `motiv` object.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

`setFilter`, `split`, `combine`

Examples

```
#####Database and Scores#####  
path <- system.file(package="MotIV")  
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))  
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))  
  
#####Input#####  
data(FOXA1_rGADEM)  
motifs <- getPWM(gadem)  
motifs.trimed <- trimPWMedge(motifs, threshold=1)  
  
#####Analysis#####  
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,  
summary(foxa1.analysis.jaspar )
```

```
#####Filters#####
f.foxal<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxal.ap1 <- f.foxal | f.ap1
foxal.filter <- filter(foxal.analysis.jaspar, f.foxal.ap1, exact=FALSE, verbose=TRUE)
foxal.split <- split(foxal.analysis.jaspar, c(f.foxal, f.ap1) , drop=FALSE, exact=FALSE,
foxal.filter.combine <- combine(foxal.filter, c(f.foxal, f.ap1), exact=FALSE, name=c("FOX
```

generateDBScores *Database Scores Functions*

Description

This functions are used to generate scores of a PWM database.

Usage

```
generateDBScores(inputDB, cc="PCC", align="SWU", nRand=1000, go=1, ge=0.5)
readDBScores(file)
writeDBScores(x, file)
```

Arguments

inputDB	A list of PWM corresponding to the database.
cc	The metric name to be used :
align	The Alignment method to be used.
go	Gap open penalty.
ge	Gap extension penalty.
nRand	The number of random PWM to be generated. The more higher it is, the more accurate score will be.
file	A character string naming a file.
x	A numeric matrix corresponding to a score.

Details

The score reflects the biases of the database. It is used to compute more precisely e-value alignments.

`generateDBScores` : Based on database properties (suchs as length, zero rate, invariant columns), `nRand` matrix are generated. A score is calculated for each matrix length with the specified alignment method and metric.

The score is associated to a database and a alignment method and metric so you don't have to generate it each time you use the same database. Use the `writeDBScores` and `readDBScores` instead. `readDBScores` : Read a score file. `writeDBScores` : Write a score file.

Value

A numeric matrix. Columns correspond respectively to the first matrix length, second matrix length, variance, mean, matrix number, distance min and max.

Warning

Because of each matrix is compare to each other, computing time is exponential. You should be aware of this fact before provided a high nRand. 5000 is a good time/accuracy rate choice.

Author(s)

Shaun Mahony, modified by Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

References

Sandelin,A. and Wasserman,W.W. (2004) Constrained binding site diversity within families of transcription factors enhances pattern discovery bioinformatics. J. Mol. Biol., 338, 207/215.

See Also

'readDBScores', 'writeDBScores'

Examples

```
#####Database and Scores#####  
path <- system.file(package="MotIV")  
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))  
#jaspar.scores <- generateDBScores(inputDB=jaspar, cc="PCC", align="SWU", nRand=1000)  
#writeDBScores(jaspar.scores, paste(path, "/extdata/jaspar_PCC_SWU.scores", sep=""))  
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))
```

getGademPWM

Recover PWM

Description

This function selects the PWMs contained in an object of type gadem.

Usage

```
getGademPWM(y)
```

Arguments

y A gadem object.

Value

A list of PWM.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
# motifs <- getGademPWM(gadem) #deprecated
motifs <- getPWM(gadem)
motifs.trimmed <- trimPWMedge(motifs, threshold=1)
```

jaspar2010

Jaspar 2010 Database

Description

Jaspar database and Jaspar score.

Usage

```
jaspar
jaspar.scores
```

Details

Jaspar is a well-known transcription factor database. Version 2010 contents 130 non-redundant matrix of TF binding sites.

The jaspar scores have been computed with Pearson Correlation Coefficient and Smith-Waterman Ungapped alignments.

Source

<http://jaspar.genereg.net/>

References

Albin Sandelin, Wynand Alkema, Pär Engström, Wyeth W. Wasserman and Boris Lenhard, JASPAR: an open-access database for eukaryotic transcription factor binding profiles, *Nucleic Acids Research*(2003)

See Also

generateDBscores, motifMatch

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
data(jaspar2010)
data(jaspar2010_scores)

#####Input#####
data(FOXAl_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxal.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,
summary(foxal.analysis.jaspar )
```

makePWM

*Constructing a pwm object***Description**

This function constructs an object of class `pwm` from a matrix. It checks that the matrix has correct dimensions and that columns add up to 1.0.

Usage

```
makePWM(pwm, alphabet="DNA")
```

Arguments

<code>pwm</code>	Matrix representing the position weight matrix
<code>alphabet</code>	Character the alphabet making up the sequence. Currently, only "DNA" is supported.

Value

An object of class `pwm`.

Author(s)

Oliver Bombom, <bombom@berkeley.edu>

Examples

```
#mFile <- system.file("Exfiles/pwm1", package="seqLogo")
#m <- read.table(mFile)
#pwm <- makePWM(m)
```

matches-class *Class "matches"*

Description

This object contains the name of the input motif and all the matches found.

Objects from the Class

Objects can be created by calls of the form `new("matches", name, aligns, similarity, valid)`.

Slots

name Motif name.

aligns Alignments found by `motifMatch`.

similarity The optional name given to the motif.

valid The alignment that should be considered as valid.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

`motiv`, `alignments`, `tf`

Examples

```
showClass("matches")
```

motifMatch *Motifs Matches Analysis*

Description

Search for motifs matches corresponding to PWM.

Usage

```
motifMatch(inputPWM, database=jaspar, DBscores=jaspar.scores, cc="PCC", align=
```

Arguments

inputPWM	A list of PWM.
database	A list of PWM corresponding to the database.
DBscores	A matrix object containing the scores associated to the database.
cc	The metric name to be used
align	The Alignment method to be used.
top	The number of identified transcription factors per motif.
go	Gap open penalty.
ge	Gap extension penalty.

Details

For a set of PWMs given by `inputPWM`, this function realizes alignments with each motif of the database and returns the `top` best motifs. If no `database` is provided, the function will use `jaspar` by loading `data(jaspar2010)`. If no `DBscores` is given, `jaspar.scores` from `data(jaspar2010_scores)` will be used.

The e-value is computed according the metric name `cc` and is corrected by the `DBscores`.

Value

A `motiv` object.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

References

S Mahony, PE Auron, PV Benos, DNA familial binding profiles made easy: comparison of various motif alignment and clustering strategies, *PLoS Computational Biology* (2007) 3(3):e61

See Also

`generatedDBScores`

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )
```

 occurrences

Motifs Occurences and Co-occurences

Description

Get the number of motifs occurrences and co-occurrences from a rGADEM object.

Usage

```
occurrences (gadem)
cooccurrences (x)
```

Arguments

gadem	An object of type rGADEM.
x	A contingency table.

Value

occurrences returns the contingency table of the number of motifs per sequences.

This object can be put in cooccurrences to return the number of sequences where two motifs appear together.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#TODO
```

 motiv-class

Class "motiv"

Description

This object contains all informations about the motiv analysis.

Objects from the Class

Objects can be created by calls of the form `new("motiv", input, bestMatch, argv)`.

Slots

input List of input PWM.

bestMatch Object of class "matches".

argv List of arguments used.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

matches, alignments, transcriptionFactor

Examples

```
showClass("motiv")
```

motiv-methods

Motiv methods

Description

Methods for motiv objects.

Usage

```
## S4 method for signature 'motiv':
summary(object)
## S4 method for signature 'motiv':
names(x)
## S4 method for signature 'motiv':
length(x)
## S4 method for signature 'motiv':
similarity(x)
## S4 method for signature 'motiv':
[(x,i,j=ANY, bysim=TRUE, ...,exact=TRUE, drop=FALSE)
```

Arguments

object	An object of class motiv.
x	An object of class motiv.
i	A string representing a motif name.
j	NOT USED.
bysim	If TRUE, select similarity name.
...	Further potential arguments passed to methods.
exact	If TRUE, search only for perfect name match.
drop	If TRUE, no match motifs will be dropped.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

matches, alignments, tf

Examples

```
showClass("motiv")
```

plot-methods

Plot Motiv

Description

This functions are used to vizualise and validate `motiv` analysis.

Usage

```
## S4 method for signature 'motiv,ANY':
plot(x, y=NULL, main=NULL, sub=NULL, ncol=0, nrow=0, top=3, bysim=TRUE, rev=FALSE)
## S4 method for signature 'motiv,gadem':
plot(x, y, sort=FALSE, group=FALSE, main=NULL, sub=NULL, ncol=0, nrow=0, xlim=NUM)
```

Arguments

<code>x</code>	An object of class <code>motiv</code> .
<code>y</code>	The GADEM type object associated with the <code>motiv</code> object.
<code>ncol</code>	The desired number of columns.
<code>nrow</code>	The desired number of rows.
<code>top</code>	The top matches to be print.
<code>rev</code>	Print reverse motif for negatif strand.
<code>main</code>	A title of the plot.
<code>sub</code>	A sub title of the plot.
<code>type</code>	The type of plotting.
<code>strand</code>	If TRUE, two distributions will be plot for forward and reverse strand.
<code>group</code>	If TRUE, similar motif will be group.
<code>sort</code>	If TRUE, motifs will be plot by decreased variance order.
<code>bysim</code>	If TRUE, the 'similar' field (defined with the <code>combine</code> function) will be print instead of the original name.
<code>xlim</code>	numeric vectors of length 2, giving the x coordinates ranges.
<code>correction</code>	If TRUE, corrects the position according to the alignment.
<code>harg, darg, carg, varg</code>	Parameters for histograms, densities, curves and venn diagrams.
<code>...</code>	Further potential arguments passed to methods.

Details

A single `motiv` object (usually provided by `motifMatch`) will plot the list of identified transcription factors for each motif. With `rev=TRUE`, the transcription factor logo will be printed to correspond to the real alignment instead of original TF PWM.

Giving a `motiv` object and a `gadem` object with `type="distribution"` will show the motif repartition within `gadem` peaks. If `strand=TRUE`, a distinct distribution is made for forward and reverse strand.

A `var.test` is automatically made to help to distinguish centered distribution. The distribution with lowest variance is assigned as "reference" distribution to compute the `var.test` statistic. With `sort=TRUE`, distributions are plotted according to decreasing statistics.

`type="distance"` indicates to compute and plot the distance between each pair of motifs. It also provides a Venn diagram that returns the proportion of common sequences per pair of motifs.

The `group` argument indicates to consider similar motifs as a single motif.

With `correction=TRUE` the motif position is corrected according to the alignment. It means that the gap/"N" contained in the alignments are removed to give a corrected start and end position.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)
#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.split <- split(foxa1.analysis.jaspar, c(f.foxa1, f.ap1) , drop=FALSE, exact=FALSE,
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"))

#####Plots#####
plot(foxa1.filter.combine, ncol=2, top=5, rev=FALSE, main="FOXA", bysim=TRUE)
plot(foxa1.filter.combine , gadem, ncol=2, type="distribution", correction=TRUE, group=FALSE)
plot(foxa1.filter.combine , gadem, type="distance", correction=TRUE, group=TRUE, bysim=TRUE)
```

readGademPWMfile *Read Gadem File*

Description

This function is use to read a gadem file containing PWM.

Usage

```
readGademPWMfile(file)
```

Arguments

file File's name.

Details

This function is made to read typically output file from Gadem (v1.2). Standard name is 'observed-PWMs.txt'.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)
```

readPWMfile *Read Transfac File*

Description

This function is use to read standard Transfac type file.

Usage

```
readPWMfile(file)
```

Arguments

file Transfac file's name.

Details

This function is designed to read standard Transfac type file. For more information about the format, please refer to <http://mcast.sdsc.edu/doc/transfac-format.html>

Value

A list of matrix.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))
```

seqLogo

Plot a sequence logo for a given position weight matrix

Description

This function takes the 4xW position weight matrix of a DNA sequence motif and plots the corresponding sequence logo.

Usage

```
seqLogo(pwm, ic.scale=TRUE, xaxis=TRUE, yaxis=TRUE, xfontsize=15, yfontsize=15)
```

Arguments

pwm	numeric	The 4xW position weight matrix.
ic.scale	logical	If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
xaxis	logical	If TRUE, an X-axis will be plotted.
yaxis	logical	If TRUE, a Y-axis will be plotted.
xfontsize	numeric	Font size to be used for the X-axis.
yfontsize	numeric	Font size to be used for the Y-axis.
vmargins	numeric	Vertical margins.
hmargins	numeric	Horizontal margins.
size	numeric	Graphic size.

Details

Within each column, the height of a given letter is proportional to its frequency at that position. If ic.scale is TRUE, the height of each column in the plot indicates the information content at that position of the motif. Otherwise, the height of all columns are identical.

Value

None.

Author(s)

Oliver Bombom, <bombom@berkeley.edu>

Examples

```
#mFile <- system.file("Exfiles/pwm1", package="seqLogo")
#m <- read.table(mFile)
#pwm <- makePWM(m)
#seqLogo(pwm)
```

setFilter

Set Motif Filter

Description

This function is use to set a motif filter.

Usage

```
setFilter(name="", tfname="", evalueMax=1, top=10, lengthMax=100, valid=NULL)
```

Arguments

name	A name or a list of names.
tfname	A transcription factor name or a list of TF names.
evalueMax	An evalue between 0 and 1.
top	Defines the depth of the filter.
lengthMax	The maximum motif length.
valid	The alignment that should be considered as valid.

Value

A filter object.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

filter, split, combine

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.split <- split(foxa1.analysis.jaspar, c(f.foxa1, f.ap1) , drop=FALSE, exact=FALSE,
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOX
```

split-methods

*Split Motiv Object***Description**

This function splits a 'motiv' object according filters.

Usage

```
## S4 method for signature 'motiv, filters':
split(x, f, exact=FALSE, drop=FALSE, verbose=TRUE, ...)
```

Arguments

x	An object of class <code>motiv</code> (usually provided by <code>motifMatch</code>).
f	A filter or a set of filter for <code>motiv</code> object.
drop	If <code>TRUE</code> , no match motifs will be dropped.
verbose	If <code>FALSE</code> , no output will be print.
exact	If <code>TRUE</code> , search only for perfect name match.
...	Further potential arguments passed to methods.

Details

This function is used to split motifs that correspond to the filters.

Many `filter` could be pass in argument separated by coma. They will be considered independently (coma is considered as OR).

Value

A list of `motiv` object.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

`setFilter`, `filter`, `combine`

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evaluateMax=10^-5)
f.ap1 <- setFilter (tfname="AP1", top=3)
f.foxa1.ap1 <- f.foxa1 | f.ap1
foxa1.filter <- filter(foxa1.analysis.jaspar, f.foxa1.ap1, exact=FALSE, verbose=TRUE)
foxa1.split <- split(foxa1.analysis.jaspar, c(f.foxa1, f.ap1) , drop=FALSE, exact=FALSE,
foxa1.filter.combine <- combine(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOX
```

transcriptionFactor-class

Transcription Factor Class

Description

This object contains the Transcription Factor name and PWM.

Objects from the Class

Objects can be created by calls of the form `new("transcriptionFactor", name, pwm)`.

Slots

name TF name.

pwm TF PWM.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

motiv, matches, alignments

Examples

```
showClass("transcriptionFactor")
```

trimPWMedge	<i>Trim PWM edge</i>
-------------	----------------------

Description

This function is use to cut edges with low information content.

Usage

```
trimPWMedge(x, threshold=1)
```

Arguments

`x` A matrix representing a PWM.
`threshold` A transcription factor name or a list of TF names.

Value

A PWM.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

See Also

makePWM

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimmed <- trimPWMedge(motifs, threshold=1)
```

```
viewAlignments      Print Motifs Alignments
```

Description

This function return a list of the alignments of a motif object for each motif.

Usage

```
viewAlignments(x)
```

Arguments

x An object of class motif (usually provided by motifMatch).

Details

This function shows the alignments for each motif.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
summary(foxa1.analysis.jaspar )
viewAlignments(foxa1.analysis.jaspar )
```

```
viewMotifs-methods Print Identified Motifs
```

Description

This function return a list of the identified motifs contained in a motif object.

Usage

```
## S4 method for signature 'motif':
viewMotifs(x, n=100)
```


Arguments

- x An object of class `motiv` (usually provided by `motifMatch`).
- n The number of motifs shown.

Details

This function shows the number of identified motifs.

Value

A list of motifs names.

Author(s)

Eloi Mercier <<eloi.mercier@ircm.qc.ca>>

Examples

```
#####Database and Scores#####
path <- system.file(package="MotIV")
jaspar <- readPWMfile(paste(path, "/extdata/jaspar2010.txt", sep=""))
jaspar.scores <- readDBScores(paste(path, "/extdata/jaspar2010_PCC_SWU.scores", sep=""))

#####Input#####
data(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimmed <- trimPWMedge(motifs, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar,
viewMotifs(foxa1.analysis.jaspar, 5)
```

Index

*Topic classes

- alignments-class, 1
- filter-class, 5
- filters-methods, 6
- matches-class, 12
- motiv-class, 14
- motiv-methods, 15
- transcriptionFactor-class, 22

*Topic methods

- combine, 2
- filter, 7
- plot-methods, 16
- split-methods, 21

*Topic misc

- exportAsRangedData, 3
- exportAsTransfacFile, 4
- generateDBScores, 8
- getGademPWM, 9
- jaspar2010, 10
- makePWM, 11
- motifMatch, 12
- occurrences, 14
- readGademPWMFile, 18
- readPWMfile, 18
- seqLogo, 19
- setFilter, 20
- trimPWMedge, 23
- viewAlignments, 24
- viewMotifs-methods, 24
- [, motiv-method (*motiv-methods*), 15
- &, filters, filters-method (*filters-methods*), 6

alignments-class, 1

- combine, 2
- combine, motiv, filters-method (*combine*), 2
- combine, motiv, list-method (*combine*), 2
- combine-method (*combine*), 2
- cooccurrences (*occurrences*), 14

exportAsRangedData, 3

exportAsTransfacFile, 4

- filter, 7
- filter, motiv, filters-method (*filter*), 7
- filter, motiv, list-method (*filter*), 7
- filter-class, 5
- filter-methods (*filter*), 7
- filters-methods, 6

- generateDBScores, 8
- getGademPWM, 9

- jaspar (*jaspar2010*), 10
- jaspar2010, 10

- length, motiv-method (*motiv-methods*), 15

- makePWM, 11
- matches-class, 12
- motifMatch, 12
- motiv (*motiv-methods*), 15
- motiv-class, 14
- motiv-methods, 15

- names, motiv-method (*motiv-methods*), 15
- names<-, motiv-method (*motiv-methods*), 15

occurrences, 14

- plot, motiv, ANY-method (*plot-methods*), 16
- plot, motiv, gadem-method (*plot-methods*), 16
- plot, motiv-method (*plot-methods*), 16
- plot-methods, 16

- readDBScores (*generateDBScores*), 8
- readGademPWMFile, 18
- readPWMfile, 18

seqLogo, [19](#)
setFilter, [20](#)
show, filter-method
 (*filters-methods*), [6](#)
show, motiv-method
 (*motiv-methods*), [15](#)
similarity (*motiv-methods*), [15](#)
similarity, list-method
 (*motiv-methods*), [15](#)
similarity, motiv-method
 (*motiv-methods*), [15](#)
split, motiv, filters-method
 (*split-methods*), [21](#)
split, motiv, list-method
 (*split-methods*), [21](#)
split-methods, [21](#)
summary, filter-method
 (*filters-methods*), [6](#)
summary, filters-method
 (*filters-methods*), [6](#)
summary, list-method
 (*motiv-methods*), [15](#)
summary, motiv-method
 (*motiv-methods*), [15](#)

transcriptionFactor-class, [22](#)
trimPWMedge, [23](#)

viewAlignments, [24](#)
viewMotifs (*viewMotifs-methods*),
 [24](#)
viewMotifs, motiv-method
 (*viewMotifs-methods*), [24](#)
viewMotifs-methods, [24](#)

writeDBScores (*generateDBScores*),
 [8](#)