

# Package ‘MotIV’

October 9, 2015

**Type** Package

**Title** Motif Identification and Validation

**Version** 1.24.0

**Date** 2014-04-09

**Author** Eloi Mercier, Raphael Gottardo

**Maintainer** Eloi Mercier <emercier@bcgsc.ca>, Raphael Gottardo  
<rgottard@fhcrc.org>

**Depends** R (>= 2.10), BiocGenerics (>= 0.1.0)

**Imports** graphics, grid, methods, IRanges (>= 1.13.5), Biostrings (>= 1.24.0), lattice, rGADEM, stats, utils

**Suggests** rtracklayer

**Description** This package makes use of STAMP for comparing a set of motifs to a given database (e.g. JASPAR). It can also be used to visualize motifs, motif distributions, modules and filter motifs.

**License** GPL-2

**biocViews** Microarray, ChIPchip, ChIPSeq, GenomicSequence, MotifAnnotation

**NeedsCompilation** yes

## R topics documented:

alignments-class . . . . .	2
as.data.frame . . . . .	3
combineMotifs . . . . .	4
exportAsRangedData . . . . .	5
exportAsTransfacFile . . . . .	6
filter . . . . .	7
filter-class . . . . .	9
filters-methods . . . . .	10
FOXA1_rGADEM . . . . .	10

generateDBScores . . . . .	11
getGademPWM . . . . .	13
getPWM . . . . .	13
jaspar2010 . . . . .	14
makePWM . . . . .	15
matches-class . . . . .	16
motifDistances . . . . .	16
motifMatch . . . . .	17
motiv-class . . . . .	19
motiv-methods . . . . .	20
occurences . . . . .	21
plot-methods . . . . .	21
readGademPWMFile . . . . .	24
readPWMfile . . . . .	24
seqLogo2 . . . . .	25
setFilter . . . . .	26
split-methods . . . . .	27
transcriptionFactor-class . . . . .	29
trimPWMedge . . . . .	29
viewAlignments . . . . .	30
viewMotifs-methods . . . . .	31

**Index** **33**

---

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, evaluate, sequence, match, strand)`.

### Slots

**TF** Object of class "TF"  
**evaluate** The e-value of the alignment.  
**sequence** The input sequence aligned.  
**match** The TF sequence wich as been matched.  
**strand** The strand of the alignment.

### Author(s)

Eloi Mercier <<emercier@chibi.ubc.ca>>

**See Also**

motiv, matches, transcriptionFactor

---

as.data.frame

*Coerce to a Data Frame*

---

**Description**

This function coerces a MotIV object into a data frame.

**Usage**

```
## S4 method for signature 'motiv'  
as.data.frame(x)
```

**Arguments**

x                    An object of class motiv.

**Details**

'as.data.frame' returns a data frame.

This object regroups all the TF identified by MotIV with the corresponding evaluate and alignments.

**Value**

A data.frame object.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.ca>>

**See Also**

data.frame, viewAlignments

**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 <- lapply(motifs, trimPWMedge, threshold=1)
```

```
#####Analysis#####  
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,DBscores=jaspar.scores  
foxa1.analysis.jaspar.df = as.data.frame(foxa1.analysis.jaspar)  
head(foxa1.analysis.jaspar.df)
```

---

combineMotifs

*Combine Motifs*

---

## Description

This function combines motifs according to a set of filters.

## Usage

```
## S4 method for signature 'motiv,filters'  
combineMotifs(x, y, name=NULL,exact=TRUE,verbose=TRUE)
```

## Arguments

x	An object of class <code>motiv</code> .
y	A filter or a set of filter.
name	Name(s) to be given for similar motifs.
verbose	If FALSE, no output will be print.
exact	If TRUE, 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 <<emercier@chibi.ubc.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.trimmed <- lapply(motifs, trimPWMedge, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar, DBscores=jaspar.scores)
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 <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb
```

---

exportAsRangedData      *Export MotIV Results*

---

**Description**

Export your

**Usage**

```
exportAsRangedData(x, y, 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.
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 <<emercier@chibi.ubc.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 <- lapply(motifs,trimPWMedge, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,DBscores=jaspar.scores)
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, verbose=TRUE)
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb

#####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=FALSE, bysim=TRUE, strand
#plot(foxa1.filter.combine ,gadem,type="distance", correction=TRUE, group=TRUE, bysim=TRUE, main="FOXA1", strand

#####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**

```
## S4 method for signature 'motiv'
exportAsTransfacFile(x, file)
## S4 method for signature 'list'
exportAsTransfacFile(x, file)
```

**Arguments**

x                    An object of class `motiv` to be export.  
 file                A character string naming a file.

**Details**

This function is made to provide 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 refer to <http://www.benoslab.pitt.edu/stamp/help.html>.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.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 <- lapply(motifs, trimPWMedge, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar, DBscores=jaspar.scores)
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, verbose=TRUE)
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb

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

---

 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**

x	An object of class <code>motiv</code> .
f	A filter or a set of filter for <code>motiv</code> object.
verbose	If FALSE, no output will be print.
exact	If TRUE, 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 <<emercier@chibi.ubc.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 <- lapply(motifs,trimPWMedge, threshold=1)

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



```
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb
```

---

filter-class	Class "filter"
--------------	----------------

---

## Description

This object information to be apply as filter.

## Details

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

## Objects from the Class

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

## Slots

**name** A name or a list of names.

**tfname** A transcription factor name or a list of TF names.

**evalueMax** 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 <<emercier@chibi.ubc.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)  
## S4 method for signature 'filter'  
names(x)  
## S4 method for signature 'filters'  
names(x)
```

### Arguments

object	An object of class filter.
x	An object of class filter.

### Author(s)

Eloi Mercier <<emercier@chibi.ubc.ca>>

### See Also

setFilter , filter, split, combine

### Examples

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

---

FOXA1\_rGADEM

*Dataset for FOXA1*

---

### Description

This dataset contains results obtained by rGADEM for the FOXA1 transcription factor.

### Usage

gadem

## References

<http://genomebiology.com/2008/9/9/R137>

## Examples

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

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

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

---

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 higer 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 bias of the database. It is used to compute more precisely e-value alignments.

`generateDBScores` : Based on database properties (such 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 an 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 each matrix is compared to each other, computing time is exponential. You should be aware of this fact before providing a high nRand. 5000 is a good time/accuracy rate choice.

**Author(s)**

Shaun Mahony, modified by Eloi Mercier <<emercier@chibi.ubc.ca>>

**References**

Sandelin, A. and Wasserman, W.W. (2004) Constrained binding site diversity within families of transcription factors. *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	<i>Recover PWM</i>
-------------	--------------------

---

**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 <<emercier@chibi.ubc.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.trimed <- lapply(motifs, trimPWMedge, threshold=1)
```

---

getPWM	<i>Get PWMs from a motiv object</i>
--------	-------------------------------------

---

**Description**

Get PWMs from a motiv object.

**Usage**

```
## S4 method for signature 'motiv'
getPWM(x)
```

**Arguments**

x                    An object of class `motif`.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.ca>>

---

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(FOXA1_rGADEM)
motifs <- getPWM(gadem)
motifs.trimmed <- lapply(motifs,trimPWMedge, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,DBscores=jaspar.scores)
summary(foxa1.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 Bembom, <bembom@berkeley.edu>

**Examples**

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

---

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

---

### 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 <<emercier@chibi.ubc.ca>>

### See Also

`motiv`, `alignments`, `tf`

### Examples

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

---

motifDistances	<i>Clustering PWMs Computation</i>
----------------	------------------------------------

---

### Description

Set of functions to perform clustering of PWMs.

### Usage

```
motifDistances(inputPWM, DBscores=jaspar.scores, cc="PCC", align="SWU", top=5, go=1, ge=0.5)
motifHclust(x,...)
motifCutree(tree,k=NULL, h=NULL)
```



**Arguments**

inputPWM, DBscores, cc, align, top, go, ge  
   Option for the PWMs distances computation. Refere to motifMatch.  
 x, ...                                  Arguments to pass to the hclust function. See hclust.  
 tree, k, h                              Arguments to pass to the cutree function. See cutree.

**Details**

This function are made to perform motifs clustering.

The ‘motifDistances’ function computes the distances between each pair of motifs using the specified alignment.

The ‘motifHclust’ and ‘motifCutree’ functions are simple redefinition of ‘hclust’ and ‘cutree’.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.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 <- lapply(motifs, trimPWMedge, threshold=1)

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

#####Clustering#####
d <- motifDistances(getPWM(foxa1.analysis.jaspar))
hc <- motifHclust(d)
plot(hc)
f <- motifCutree(hc, k=2)
foxa1.combine <- combineMotifs(foxa1.analysis.jaspar, f, exact=FALSE, name=c("Group1", "Group2"), verbose=TRUE)
```

---

motifMatch

*Motifs Matches Analysis*

---

**Description**

Search for motifs matches corresponding to PWM.

**Usage**

```
motifMatch(inputPWM, database=jaspar, DBscores=jaspar.scores, cc="PCC", align="SWU", top=5, go=1, g
```

**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 motif object.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.ca>>

**References**

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

**See Also**

generateDBScores

**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 <- lapply(motifs,trimPWMedge, threshold=1)

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

---

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 <<emercier@chibi.ubc.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, ignore.case=FALSE, drop=FALSE]
```

## Arguments

object	An object of class <code>motiv</code> .
x	An object of class <code>motiv</code> .
i	A string representing a motif name.
j	NOT USED.
bysim	If TRUE, select by similarity name.
...	Further potential arguments passed to methods.
ignore.case	if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching
exact	If TRUE, search only for perfect name match.
drop	If TRUE, no match motifs will be dropped.

## Author(s)

Eloi Mercier <<emercier@chibi.ubc.ca>>

## See Also

`matches`, `alignments`, `tf`

## Examples

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

---

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 <<emercier@chibi.ubc.ca>>

### Examples

```
data("FOXA1_rGADEM")
oc <- occurrences (gadem)
coc <- cooccurrences (oc)
coc
```

---

plot-methods

*Plot Motiv*

---

### Description

This functions are used to vizualise and validate motif 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, trim=0.05, cex)

## S4 method for signature 'motiv,gadem'
plot(x, y, sort=FALSE, group=FALSE, main=NULL, sub=NULL, ncol=0, nrow=0, xlim=NULL, correction=TRUE,
```

**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, nrow</code>	A numeric value giving the the number of columns and rows to plot.
<code>top</code>	A numeric value giving the number of best matches per motif to display.
<code>rev</code>	A logical value. If TRUE, print reverse motif for negatif strand.
<code>main</code>	An overall title for the plot: see <code>title</code> .
<code>sub</code>	A sub title for the plot: see <code>'title'</code>
<code>type</code>	What type of plot should be drawn. Possible values are : distribution to display the binding sites distribution within the peaks or distance to show the pairwise distance between motifs.
<code>strand</code>	If TRUE, distribution will be plot for both forward and reverse strand.
<code>group</code>	If TRUE, similar motifs will be grouped.
<code>sort</code>	If TRUE, motifs will be plot according their computed variance.
<code>bysim</code>	If TRUE, the <code>'similar'</code> field (defined with the <code>combine</code> function) will be print instead of the original name.
<code>xlim</code>	A numeric vectors of length 2, giving the x coordinates ranges.
<code>correction</code>	If TRUE, corrects the position according to the alignment.
<code>trim</code>	A numeric value. Define the mimimun information content value for which the logo letters are shown.
<code>col, border, lwd, lty</code>	Define respectively the color, the border, the line wide and the line type of both curve and histogram. See <code>'par'</code> .
<code>nclass</code>	A numerical value giving the number of class for the histogram.
<code>bw</code>	he smoothing bandwidth to be used to calculate the density. See <code>density</code> .
<code>cex, vcol</code>	A numerical value giving the amount by which plotting text should be magnified relative to the default.

**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 print 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 assign as "reference" distribution to compute the `var.test` statistic. With `sort=TRUE`, distribution are plot according decreasing statistic.

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

The `group` argument indicates to consider similar motif 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 <<emercier@chibi.ubc.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 <- lapply(motifs,trimPWMedge, threshold=1)
#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs,align="SWU",cc="PCC",database=jaspar,DBscores=jaspar.scores)
summary(foxa1.analysis.jaspar )

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, valueMax=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, verbose=TRUE)
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb

#####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, bysim=TRUE, strand=
plot(foxa1.filter.combine ,gadem,type="distance", correction=TRUE, group=TRUE, bysim=TRUE, main="FOXA", strand=
```

---

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 <<emercier@chibi.ubc.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 <- lapply(motifs, trimPWMedge, 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 <<emercier@chibi.ubc.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=""))
```

---

seqLogo2

---

*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**

```
seqLogo2(pwm, ic.scale=TRUE, xaxis=TRUE, yaxis=TRUE, xfontsize=15, yfontsize=15, vmargins=c(0,0), hmargins=c(0,0))
```

**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.
trim	numeric Print nucleotide only if the information content is superior to this trim threshold.

### 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.

This is an internal function for the package `MotIV`. User should prefer the `seqLogo` function from the package `seqLogo` to visualize individual motif.

### Value

None.

### Author(s)

Oliver Bombom, <bombom@berkeley.edu>, modified by Eloi Mercier <<emercier@chibi.ubc.ca>>

### Examples

```
#mFile <- system.file("Exfiles/pwm1", package="seqLogo")
#m <- read.table(mFile)
#pwm <- makePWM(m)
#seqLogo2(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 <<emercier@chibi.ubc.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.trimmed <- lapply(motifs, trimPWMedge, threshold=1)

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

#####Filters#####
f.foxa1<-setFilter(name="", tfname="FOXA1", top=3, evalueMax=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, verbose=TRUE)
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb
```

---

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=TRUE, drop=FALSE, verbose=TRUE, ...)
```

**Arguments**

x	An object of class <code>motif</code> (usually provided by <code>motifMatch</code> ).
f	A filter or a set of filter for <code>motif</code> object.
drop	If TRUE, no match motifs will be dropped.
verbose	If FALSE, no output will be printed.
exact	If TRUE, 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 passed in argument separated by comma. They will be considered independently (comma is considered as OR).

**Value**

A list of `motif` object.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.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.trimmed <- lapply(motifs, trimPWMedge, threshold=1)

#####Analysis#####
foxa1.analysis.jaspar <- motifMatch(inputPWM=motifs, align="SWU", cc="PCC", database=jaspar, DBscores=jaspar.scores)
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, verbose=TRUE)
foxa1.filter.combine <- combineMotifs(foxa1.filter, c(f.foxa1, f.ap1), exact=FALSE, name=c("FOXA1", "AP1"), verb
```

---

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 <<emercier@chibi.ubc.ca>>

**See Also**

`motiv`, `matches`, `alignments`

**Examples**

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

---

`trimPWMedge`                    *Trim PWM edge*

---

**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 <<emercier@chibi.ubc.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 <- lapply(motifs, trimPWMedge, threshold=1)
```

---

viewAlignments

*Print Motifs Alignments*

---

**Description**

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

**Usage**

```
viewAlignments(x)
```

**Arguments**

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

**Details**

This function shows the alignments for each motif.

**Author(s)**

Eloi Mercier <<emercier@chibi.ubc.ca>>

**See Also**

as.data.frame

## 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 <- lapply(motifs, trimPWMedge, threshold=1)

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

## Usage

```
## S4 method for signature 'motiv'
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 motif.

## Value

A list of motifs names.

## Author(s)

Eloi Mercier <<emercier@chibi.ubc.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 <- lapply(motifs,trimPWMedge, threshold=1)

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



# Index

- \*Topic **classes**
  - alignments-class, 2
  - filter-class, 9
  - filters-methods, 10
  - matches-class, 16
  - motiv-class, 19
  - transcriptionFactor-class, 29
- \*Topic **methods**
  - as.data.frame, 3
  - combineMotifs, 4
  - filter, 7
  - motiv-methods, 20
  - plot-methods, 21
  - split-methods, 27
- \*Topic **misc**
  - exportAsRangedData, 5
  - exportAsTransfacFile, 6
  - FOXA1\_rGADEM, 10
  - generateDBScores, 11
  - getGademPWM, 13
  - getPWM, 13
  - jaspar2010, 14
  - makePWM, 15
  - motifDistances, 16
  - motifMatch, 17
  - occurrences, 21
  - readGademPWMFile, 24
  - readPWMfile, 24
  - seqLogo2, 25
  - setFilter, 26
  - trimPWMedge, 29
  - viewAlignments, 30
  - viewMotifs-methods, 31
- [,motiv,ANY,ANY-method (motiv-methods), 20
- [,motiv-method (motiv-methods), 20
- &,filters,filters-method (filters-methods), 10
- alignments-class, 2
- as.data.frame, 3
- as.data.frame,motiv-method (as.data.frame), 3
- as.data.frame-methods (as.data.frame), 3
- combineMotifs, 4
- combineMotifs,motiv,filters-method (combineMotifs), 4
- combineMotifs,motiv,list-method (combineMotifs), 4
- combineMotifs-method (combineMotifs), 4
- cooccurrences (occurrences), 21
- exportAsRangedData, 5
- exportAsTransfacFile, 6
- exportAsTransfacFile,list-method (exportAsTransfacFile), 6
- exportAsTransfacFile,motiv-method (exportAsTransfacFile), 6
- exportAsTransfacFile-methods (exportAsTransfacFile), 6
- filter, 7
- filter,motiv,filters-method (filter), 7
- filter,motiv,list-method (filter), 7
- filter-class, 9
- filter-methods (filter), 7
- filters-methods, 10
- FOXA (FOXA1\_rGADEM), 10
- FOXA1\_rGADEM, 10
- gadem (FOXA1\_rGADEM), 10
- generateDBScores, 11
- getGademPWM, 13
- getPWM, 13
- getPWM,motiv-method (getPWM), 13
- getPWM-methods (getPWM), 13
- jaspar (jaspar2010), 14
- jaspar2010, 14

- length,motiv-method (motiv-methods), 20
- makePWM, 15
- matches-class, 16
- motifCutree (motifDistances), 16
- motifDistances, 16
- motifHclust (motifDistances), 16
- motifMatch, 17
- motiv (motiv-methods), 20
- motiv-class, 19
- motiv-methods, 20
  
- names,filter-method (filters-methods), 10
- names,filters-method (filters-methods), 10
- names,motiv-method (motiv-methods), 20
- names<- ,motiv-method (motiv-methods), 20
  
- occurrences, 21
  
- plot (plot-methods), 21
- plot,motiv,ANY-method (plot-methods), 21
- plot,motiv,gadem-method (plot-methods), 21
- plot,motiv-method (plot-methods), 21
- plot-methods, 21
  
- readDBScores (generateDBScores), 11
- readGademPWMFile, 24
- readPWMfile, 24
  
- seqLogo2, 25
- setFilter, 26
- show,filter-method (filters-methods), 10
- show,motiv-method (motiv-methods), 20
- similarity (motiv-methods), 20
- similarity,list-method (motiv-methods), 20
- similarity,motiv-method (motiv-methods), 20
- split,motiv,filters-method (split-methods), 27
- split,motiv,list-method (split-methods), 27
- split-methods, 27
- summary,filter-method (filters-methods), 10
- summary,filters-method (filters-methods), 10
  
- summary,list-method (motiv-methods), 20
- summary,motiv-method (motiv-methods), 20
  
- transcriptionFactor-class, 29
- trimPWMedge, 29
  
- viewAlignments, 30
- viewMotifs (viewMotifs-methods), 31
- viewMotifs,motiv-method (viewMotifs-methods), 31
- viewMotifs-methods, 31
  
- writeDBScores (generateDBScores), 11