Package 'motifStack'

October 9, 2013

Type Package

Version 1.4.0

Date 2012-12-20

Title Plot stacked logos for single or multiple DNA, RNA and amino acid sequence

Author Jianhong Ou, Michael Brodsky, Scot Wolfe and Lihua Julie Zhu

Maintainer Jianhong Ou <jianhong.ou@umassmed.edu>

Imports grImport, grid, XML, ade4

Depends R (>= 2.15.1), methods, grImport, grid, MotIV, ade4

Suggests RUnit, BiocGenerics, MotifDb, RColorBrewer

biocViews SequenceMatching, GenomicsSequence, Visualization

Description The motifStack package is designed for graphic
representation of multiple motifs with different similarity
scores. It works with both DNA/RNA sequence motif and amino
acid sequence motif. In addition, it provides the flexibility
for users to customize the graphic parameters such as the font type and symbol colors.

License GPL (>= 2)

Lazyload yes

R topics documented:

`motifStack-package` *Plot stacked logos for single or multiple DNA, RNA and amino acid sequence*

Description

motifStack is a package that is able to draw amino acid sequence as easy as to draw DNA/RNA sequence. motifStack provides the flexibility for users to select the font type and symbol colors. motifStack is designed for graphical representation of multiple motifs.

Author(s)

Jianhong Ou and Lihua Julie Zhu

Maintainer: Jianhong Ou <jianhong.ou@umassmed.edu>

`colorset` *retrieve color setting for logo*

Description

retrieve color setting for logo

Usage

```
colorset(alphabet="DNA", colorScheme='auto')
```

Arguments

alphabet	character, 'DNA', 'RNA' or 'AA'
colorScheme	'auto', 'charge', 'chemistry', 'classic' or 'hydrophobicity' for AA, 'auto' or 'basepairing' for DNA ro RNA

Value

A character vector of color scheme

Examples

```
col <- colorset("AA", "hydrophobicity")
```

DNAmotifAlignment *align DNA motifs*

Description

align DNA motifs for plotting motifs stack

Usage

```
DNAmotifAlignment(pfms, threshold=0.4, minimalConsensus=0, rcpostfix="(RC)", revcomp=rep(TRUE, length
```

Arguments

pfms	a list of position frequency matrices, pfms must be a list of class pfm
threshold	information content cutoff threshold for useful postions
minimalConsensus	minimal length of consensus for alignment
rcpostfix	the postfix for reverse complements
revcomp	a logical vector to indicates whether the reverse complemet should be involved into alignment

Value

a list of aligned motifs

Examples

```
pcms<-readPCM(file.path(find.package("motifStack"), "extdata"),"pcm$")
pcms<-lapply(pcms,function(.ele){.ele<-.ele[,3:ncol(.ele)];rownames(.ele)<-c("A","C","G","T");.ele})
motifs<-lapply(pcms,pcm2pfm)
motifs<-lapply(names(motifs), function(.ele, motifs){new("pfm",mat=motifs[[.ele]], name=.ele)},motifs)
motifs<-DNAmotifAlignment(motifs)
```

motifCloud *plot a DNA sequence logo cloud*

Description

Plot a DNA sequence logo cloud

Usage

```
motifCloud(motifSig, rcpostfix="(RC)",
layout=c("rectangles", "cloud", "tree"),
scale=c(6, .5), rot.per=.1,
draw.box=TRUE, draw.freq=TRUE,
box.col="gray", freq.col="gray",
group.col=NULL, groups=NULL, draw.legend=FALSE)
```

Arguments

motifSig	an object of class motifSig
rcpostfix	postfix for reverse-complement motif names, default: (RC)
layout	layout of the logo cloud, rectangles, cloud or tree
scale	A vector of length 2 indicating the range of the size of the sequence logo.
rot.per	proportion sequence logo with 90 degree rotation. Only work for "cloud" layout
draw.box	draw box for each sequence logo or not
draw.freq	label frequency of each signature or not
box.col	color of box for each sequence logo
freq.col	color of frequency label
group.col	color setting for groups
groups	a named vectors of motif groups
draw.legend	draw group color legend or not

Value

none

Examples

```
if(interactive()){
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
  gsub("_FBgn[0-9]+$", "",
gsub("[^a-zA-Z0-9]","_",
```

```
 gsub("(_[0-9]+)+$", "", names(motifs)))))
motifs <- motifs[unique(names(motifs))]
pfms <- sample(motifs, 50)
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(pfms)
hc <- MotIV::motifHclust(d)
phylog <- hclust2phylog(hc)
leaves <- names(phylog$leaves)
pfms <- pfms[leaves]
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm",mat=pfms[[.ele]], name=.ele)},pfms)
motifSig <- motifSignature(pfms, phylog, groupDistance=0.1)
motifCloud(motifSig)
}
```

`motifSig-class` *Class* `"motifSig"`

Description

An object of class `"motifSig"` represents the output of function motifSignature

Objects from the Class

Objects can be created by calls of the form `new("motifSig", signature, freq, nodelist)`.

Slots

`signatures` list object of class `"pfm"`

`freq` code"numeric" signature frequency

`nodelist` list object of class `"ouNode"`

Methods

signatures `signature(object = "motifSig")` return the signatures of motifSig

frequence `signature(object = "motifSig")` return the frequency of motifSig

nodelist `signature(object = "motifSig")` return the nodelist of motifSig

motifSig-methods *"motifSig" methods*

Description

methods for motifSig objects.

Usage

```
## S4 method for signature 'motifSig'
signatures(object)
## S4 method for signature 'motifSig'
frequence(object)
## S4 method for signature 'motifSig'
nodelist(object)
```

Arguments

object	An object of class motifSig.

Methods

signatures signature(object = "motifSig") return the signatures of motifSig

frequence signature(object = "motifSig") return the frequency of motifSig

nodelist signature(object = "motifSig") return the nodelist of motifSig

motifSignature *get signatures from motifs*

Description

extract signatures from multiple motifs by distance calculated from STAMP

Usage

```
motifSignature(pfms, phylog, groupDistance, rcpostfix="(RC)",
min.freq=2, trim=0.2, families=list())
```

Arguments

pfms	a list of objects of class pfm
phylog	an object of class phylog
groupDistance	maxmal distance of motifs in the same group
rcpostfix	postfix for reverse-complement motif names, default: (RC)
min.freq	signatures with frequency below min.freq will not be plotted
trim	minimal information content for each position of signature
families	for each family, the motif number in one signature should only count as 1

Value

an Object of class motifSig

Examples

```
if(interactive()){
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
  gsub("_FBgn[0-9]+$", "",
gsub("[^a-zA-Z0-9]","_",
 gsub("(_[0-9]+)+$", "", names(motifs)))))
motifs <- motifs[unique(names(motifs))]
pfms <- sample(motifs, 50)
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(pfms)
hc <- MotIV::motifHclust(d)
phylog <- hclust2phylog(hc)
leaves <- names(phylog$leaves)
pfms <- pfms[leaves]
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm",mat=pfms[[.ele]], name=.ele)},pfms)
motifSig <- motifSignature(pfms, phylog, groupDistance=0.1)
}
```

---

motifStack *plot a DNA sequence logo stack*

Description

Plot a DNA sequence logo stack

Usage

```
motifStack(pfms, layout=c("stack", "treeview", "phylog", "radialPhylog"), ...)
```

Arguments

pfms	a list of objects of class pfm
layout	layout of the logo stack, stack, treeview or radialPhylog
...	any parameters could to pass to plotMotifLogoStack, plotMotifLogoStackWithTree, plotMotifStackWithPhylog or plotMotifStackWithRadialPhylog

Value

none

Examples

```
if(interactive()){
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
  gsub("_FBgn[0-9]+$", "",
gsub("[^a-zA-Z0-9]","_",
 gsub("(_[0-9]+)+$", "", names(motifs)))))
motifs <- motifs[unique(names(motifs))]
pfms <- sample(motifs, 50)
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm",mat=pfms[[.ele]], name=.ele)},pfms)
motifStack(pfms, "radialPhylog")
}
```

---

ouNode-class *Class* ouNode

Description

An object of class "ouNode" represents a motif node in a cluster tree

Objects from the Class

Objects can be created by calls of the form new("ouNode", left, right, parent, distl, distr, sizel, sizer).

Slots

left: character indicates the name of left leave

right: character indicates the name of right leave

parent: character indicates the name of parent node

distl: numeric indicates the distance of left leave

distr: numeric indicates the distance of right leave

sizel: numeric indicates the size of left leave

sizer: numeric indicates the size of right leave

Examples

```
new("ouNode", left="A", right="B", parent="Root", distl=1, distr=2, sizel=1, sizer=1)
```

---

pcm-class *Class* "pcm"

Description

An object of class "pcm" represents the position count matrix of a DNA/RNA/amino-acid sequence motif. The entry stores a matrix, which in row i, column j gives the counts of observing nucleotide/or amino acid i in position j of the motif.

Objects from the Class

Objects can be created by calls of the form `new("pcm", mat, name, alphabet, color, background)`.

Slots

`mat` Object of class "matrix" The position count matrix

`name` code"character" The motif name

`alphabet` "character" The sequence alphabet. "DNA", "RNA", "AA" or "others".

`color` a "character" vector. The color setting for each symbol

`background` a "numeric" vector. The background frequency.

Methods

addBlank `signature(x="pcm", n="numeric", b="logical")` add space into the position count matrix for alignment. b is a bool value, if TRUE, add space to the 3' end, else add space to the 5' end. n indicates how many spaces should be added.

coerce `signature(from = "pcm", to = "matrix")`: convert object pcm to matrix

getIC `signature(x = "pcm",)` Calculate information content profile for position frequency matrix.

matrixReverseComplement `signature(x = "pcm")` get the reverse complement of position frequency matrix.

trimMotif `signature(x = "pfm", t= "numeric")` trim motif by information content.

plot `signature(x = "pcm")` Plots the sequence logo of the position count matrix.

Examples

```
pcm <- read.table(file.path(find.package("motifStack"), "extdata", "bin_SOLEXA.pcm"))
pcm <- pcm[,3:ncol(pcm)]
rownames(pcm) <- c("A","C","G","T")
motif <- new("pcm", mat=as.matrix(pcm), name="bin_SOLEXA")
plot(motif)
```

pcm-methods *"pcm" methods*

Description

methods for pcm objects.

Usage

```
## S4 method for signature 'pcm,numeric,logical'
addBlank(x,n,b)
## S4 method for signature 'pcm,ANY'
getIC(x,p="missing")
## S4 method for signature 'pcm'
matrixReverseComplement(x)
## S4 method for signature 'pcm,ANY'
plot(x,y="missing",...)
## S4 method for signature 'pcm,ANY'
pcm2pfm(x,background="missing")
## S4 method for signature 'matrix,ANY'
pcm2pfm(x,background="missing")
## S4 method for signature 'matrix,numeric'
pcm2pfm(x,background)
## S4 method for signature 'data.frame,ANY'
pcm2pfm(x,background="missing")
## S4 method for signature 'data.frame,numeric'
pcm2pfm(x,background)
## S4 method for signature 'pcm,numeric'
trimMotif(x,t)
```

Arguments

x	An object of class pfm. For getIC, if parameter p is followed, x should be an object of matrix. For pcm2pfm, x also could be an object of matrix.
y	Not use.
p	p is the background frequency.
n	how many spaces should be added.
b	logical value to indicate where the space should be added.
background	a "numeric" vector. The background frequency.
t	numeric value of information content threshold for trimming.
...	Further potential arguments passed to plotMotifLogo.

Methods

addBlank `signature(x="pcm", n="numeric", b="logical")` add space into the position count matrix for alignment. b is a bool value, if TRUE, add space to the 3' end, else add space to the 5' end. n indicates how many spaces should be added.

coerce `signature(from = "pcm", to = "matrix")`: convert object pcm to matrix

getIC `signature(x = "pcm",)` Calculate information content profile for position frequency matrix.

matrixReverseComplement `signature(x = "pcm")` get the reverse complement of position frequency matrix.

plot `signature(x = "pcm")` Plots the sequence logo of the position count matrix.

trimMotif `signature(x = "pfm", t= "numeric")` trim motif by information content.

Examples

```
pcm <- read.table(file.path(find.package("motifStack"), "extdata", "bin_SOLEXA.pcm"))
pcm <- pcm[,3:ncol(pcm)]
rownames(pcm) <- c("A","C","G","T")
motif <- new("pcm", mat=as.matrix(pcm), name="bin_SOLEXA")
getIC(motif)
matrixReverseComplement(motif)
as(motif,"matrix")
pcm2pfm(motif)
```

---

pfm-class *Class* "pfm"

Description

An object of class "pfm" represents the position frequency matrix of a DNA/RNA/amino-acid sequence motif. The entry stores a matrix, which in row i, column j gives the frequency of observing nucleotide/or amino acid i in position j of the motif.

Objects from the Class

Objects can be created by calls of the form `new("pfm", mat, name, alphabet, color, background)`.

Slots

`mat` Object of class "matrix" The position frequency matrix

`name` code"character" The motif name

`alphabet` "character" The sequence alphabet. "DNA", "RNA", "AA" or "others".

`color` a "character" vector. The color setting for each symbol

`background` a "numeric" vector. The background frequency.

Methods

addBlank `signature(x="pfm", n="numeric", b="logical")` add space into the position frequency matrix for alignment. b is a bool value, if TRUE, add space to the 3' end, else add space to the 5' end. n indicates how many spaces should be added.

coerce `signature(from = "pfm", to = "matrix")`: convert object pfm to matrix

getIC `signature(x = "pfm",)` Calculate information content profile for position frequency matrix.

getIC `signature(x = "matrix", p = "numeric")` Calculate information content profile for matrix. p is the background frequency

matrixReverseComplement `signature(x = "pfm")` get the reverse complement of position frequency matrix.

trimMotif `signature(x = "pfm", t= "numeric")` trim motif by information content.

plot `signature(x = "pfm")` Plots the sequence logo of the position frequency matrix.

Examples

```
pcm <- read.table(file.path(find.package("motifStack"), "extdata", "bin_SOLEXA.pcm"))
pcm <- pcm[,3:ncol(pcm)]
rownames(pcm) <- c("A","C","G","T")
motif <- pcm2pfm(pcm)
motif <- new("pfm", mat=motif, name="bin_SOLEXA")
plot(motif)
```

`pfm-methods` *"pfm" methods*

Description

methods for pfm objects.

Usage

```
## S4 method for signature 'pfm,numeric,logical'
addBlank(x,n,b)
## S4 method for signature 'pfm,ANY'
getIC(x,p="missing")
## S4 method for signature 'matrix,numeric'
getIC(x,p)
## S4 method for signature 'pfm'
matrixReverseComplement(x)
## S4 method for signature 'pfm,ANY'
plot(x,y="missing",...)
## S4 method for signature 'pfm,numeric'
trimMotif(x,t)
```

Arguments

x	An object of class pfm. For getIC, if parameter p is followed, x should be an object of matrix.
y	Not use.
p	p is the background frequency.
n	how many spaces should be added.
b	logical value to indicate where the space should be added.
t	numeric value of information content threshold for trimming.
...	Further potential arguments passed to plotMotifLogo.

Methods

addBlank `signature(x="pfm", n="numeric", b="logical")` add space into the position frequency matrix for alignment. b is a bool value, if TRUE, add space to the 3' end, else add space to the 5' end. n indicates how many spaces should be added.

getIC `signature(x = "pfm",)` Calculate information content profile for position frequency matrix.

getIC `signature(x = "matrix", p = "numeric")` Calculate information content profile for matrix. p is the background frequency

matrixReverseComplement `signature(x = "pfm")` get the reverse complement of position frequency matrix.

plot `signature(x = "pfm")` Plots the sequence logo of the position frequency matrix.

trimMotif `signature(x = "pfm", t= "numeric")` trim motif by information content.

Examples

```
pcm <- read.table(file.path(find.package("motifStack"), "extdata", "bin_SOLEXA.pcm"))
pcm <- pcm[,3:ncol(pcm)]
rownames(pcm) <- c("A","C","G","T")
motif <- pcm2pfm(pcm)
motif <- new("pfm", mat=motif, name="bin_SOLEXA")
getIC(motif)
matrixReverseComplement(motif)
addBlank(motif, 1, FALSE)
addBlank(motif, 3, TRUE)
as(motif,"matrix")
```

---

plotMotifLogo *plot sequence logo*

Description

plot amino acid or DNA sequence logo

Usage

```
plotMotifLogo(pfm, motifName, p=rep(0.25, 4), font="Helvetica-Bold",
colset=c("#00811B","#2000C7","#FFB32C","#D00001"),
xaxis=TRUE,yaxis=TRUE,xlab="position",ylab="bits",
xlcex=1.2, ylcex=1.2, ncex=1.2)
```

Arguments

pfm	a position frequency matrices
motifName	motif name
p	background possibility
font	font of logo
colset	color setting for each logo letter
xaxis	draw x-axis or not
yaxis	draw y-axis or not
xlab	x-label, do nothing if set xlab as NA
ylab	y-label, do nothing if set ylab as NA
xlcex	cex value for x-label
ylcex	cex value for y-label
ncex	cex value for motif name

Value

none

Examples

```
pcm<-matrix(runif(40,0,100),nrow=4,ncol=10)
pfm<-pcm2pfm(pcm)
rownames(pfm)<-c("A","C","G","T")
plotMotifLogo(pfm)
```

---

plotMotifLogoA *plot sequence logo without plot.new*

Description

plot amino acid or DNA sequence logo in a given canvas

Usage

```
plotMotifLogoA(pfm, font="Helvetica-Bold")
```

Arguments

pfm	an object of pfm
font	font of logo

Value

none

Examples

```
pcm<-matrix(runif(40,0,100),nrow=4,ncol=10)
pfm<-pcm2pfm(pcm)
rownames(pfm)<-c("A","C","G","T")
motif <- new("pfm", mat=pfm, name="bin_SOLEXA")
plotMotifLogoA(motif)
```

---

plotMotifLogoStack *plot sequence logos stack*

Description

plot sequence logos stack

Usage

```
plotMotifLogoStack(pfms, ...)
```

Arguments

pfms	a list of position frequency matrices, pfms must be a list of class pfm
...	other parameters can be passed to plotMotifLogo function

Value

none

Examples

```
pcm1<-matrix(c(0,50,0,50,
   100,0,0,0,
   0,100,0,0,
   0,0,100,0,
   0,0,0,100,
   50,50,0,0,
   0,0,50,50), nrow=4)
pcm2<-matrix(c(50,50,0,0,
   0,100,0,0,
   0,50,50,0,
```

```
   0,0,0,100,
   50,50,0,0,
   0,0,50,50), nrow=4)
rownames(pcm1)<-c("A","C","G","T")
rownames(pcm2)<-c("A","C","G","T")
pfms<-list(p1=new("pfm",mat=pcm2pfm(pcm1),name="m1"),
   p2=new("pfm",mat=pcm2pfm(pcm2),name="m2"))
pfms<-DNAmotifAlignment(pfms)
plotMotifLogoStack(pfms)
```

`plotMotifLogoStackWithTree` *plot sequence logos stack with hierarchical cluster tree*

Description

plot sequence logos stack with hierarchical cluster tree

Usage

```
plotMotifLogoStackWithTree(pfms, hc, treewidth=1/8, trueDist=FALSE, ...)
```

Arguments

`pfms`	a list of position frequency matrices, pfms must be a list of class pfm
`hc`	an object of the type produced by stats::hclust
`treewidth`	the width to show tree
`trueDist`	logical flags to use hclust height or not.
`...`	other parameters can be passed to plotMotifLogo function

Value

none

Examples

```
#####Input#####
pcms<-readPCM(file.path(find.package("motifStack"), "extdata"),"pcm$")
pcms<-lapply(pcms,function(.ele){.ele<-.ele[,3:ncol(.ele)];rownames(.ele)<-c("A","C","G","T");.ele})
motifs<-lapply(pcms,pcm2pfm)

#####Clustering#####
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(motifs)
hc <- MotIV::motifHclust(d)

##reorder the motifs for plotMotifLogoStack
motifs<-motifs[hc$order]
```

```
motifs<-lapply(names(motifs), function(.ele, motifs){new("pfm",mat=motifs[[.ele]], name=.ele)},motifs)
##do alignment
motifs<-DNAmotifAlignment(motifs)
##plot stacks
plotMotifLogoStack(motifs, ncex=1.0)
plotMotifLogoStackWithTree(motifs, hc=hc)
```

`plotMotifStackWithPhylog` *plot sequence logo stacks with a ape4-style phylogenic tree*

Description

plot sequence logo stacks with a ape4-style phylogenic tree

Usage

```
plotMotifStackWithPhylog(phylog, pfms=NULL,
f.phylog = 0.3, f.logo = NULL, cleaves =1, cnodes =0,
labels.leaves = names(phylog$leaves), clabel.leaves=1,
labels.nodes = names(phylog$nodes), clabel.nodes = 0)
```

Arguments

`phylog`	an object of class phylog
`pfms`	a list of objects of class pfm
`f.phylog`	a size coefficient for tree size (a parameter to draw the tree in proportion to leaves label)
`f.logo`	a size coefficient for the motif
`cleaves`	a character size for plotting the points that represent the leaves, used with par("cex")*cleaves. If zero, no points are drawn
`cnodes`	a character size for plotting the points that represent the nodes, used with par("cex")*cnodes. If zero, no points are drawn
`labels.leaves`	a vector of strings of characters for the leaves labels
`clabel.leaves`	a character size for the leaves labels, used with
`labels.nodes`	a vector of strings of characters for the nodes labels
`clabel.nodes`	a character size for the nodes labels, used with par("cex")*clabel.nodes. If zero, no nodes labels are drawn

Value

none

See Also

plot.phylog

Examples

```
if(interactive()){
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
  gsub("_FBgn[0-9]+$", "",
gsub("[^a-zA-Z0-9]","_",
 gsub("(_[0-9]+)+$", "", names(motifs)))))
motifs <- motifs[unique(names(motifs))]
pfms <- sample(motifs, 50)
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(pfms)
hc <- MotIV::motifHclust(d)
phylog <- hclust2phylog(hc)
leaves <- names(phylog$leaves)
pfms <- pfms[leaves]
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm",mat=pfms[[.ele]], name=.ele)},pfms)
pfms <- DNAmotifAlignment(pfms, minimalConsensus=3)
plotMotifStackWithPhylog(phylog, pfms, f.phylog=0.3, cleaves = 0.5, clabel.leaves = 0.7)
}
```

---

plotMotifStackWithRadialPhylog

plot sequence logo stacks with a radial phylogenic tree

Description

plot sequence logo stacks with a radial phylogenic tree

Usage

```
plotMotifStackWithRadialPhylog(phylog, pfms=NULL,
circle=1, circle.motif=NA, cleaves=1, cnodes=0,
labels.leaves=names(phylog$leaves), clabel.leaves=1,
labels.nodes=names(phylog$nodes), clabel.nodes=0,
draw.box=FALSE,
col.leaves=rep("black", length(labels.leaves)),
col.leaves.bg=NULL, col.leaves.bg.alpha=1,
col.bg=NULL, col.bg.alpha=1,
col.inner.label.circle=NULL, col.outer.label.circle=NULL, outer.label.circle.width="default",
clockwise =FALSE, init.angle=if(clockwise) 90 else 0,
angle=360)
```

Arguments

phylog	an object of class phylog
pfms	a list of objects of class pfm
circle	a size coefficient for the outer circle
circle.motif	a size coefficient for the motif circle
cleaves	a character size for plotting the points that represent the leaves, used with par("cex")*cleaves. If zero, no points are drawn
cnodes	a character size for plotting the points that represent the nodes, used with par("cex")*cnodes. If zero, no points are drawn
labels.leaves	a vector of strings of characters for the leaves labels
clabel.leaves	a character size for the leaves labels, used with
labels.nodes	a vector of strings of characters for the nodes labels
clabel.nodes	a character size for the nodes labels, used with par("cex")*clabel.nodes. If zero, no nodes labels are drawn
draw.box	if TRUE draws a box around the current plot with the function box()
col.leaves	a vector of colors for leaves labels
col.leaves.bg	a vector of colors for background of leaves labels
col.leaves.bg.alpha	alpha value [0, 1] for the colors of backgroud of leaves labels
col.bg	a vector of colors for background
col.bg.alpha	a alpha value [0, 1] of colors for background
col.inner.label.circle	a vector of colors for inner cirlce of leaves labels
col.outer.label.circle	a vector of colors for outer circle of leaves labels
outer.label.circle.width	width for outer circle of leaves labels
clockwise	a logical value indicating if slices are drawn clockwise or counter clockwise
init.angle	number specifying the starting angle (in degrees) for the slices. Defaults to 0 (i.e., '3 o'clock') unless clockwise is true where init.angle defaults to 90 (degrees), (i.e., '12 o'clock')
angle	number specifying the angle (in degrees) for phylogenic tree. Defaults 360

Value

none

See Also

plot.phylog

Examples

```
if(interactive()){
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
  gsub("_FBgn[0-9]+$", "",
gsub("[^a-zA-Z0-9]","_",
 gsub("(_[0-9]+)+$", "", names(motifs)))))
motifs <- motifs[unique(names(motifs))]
pfms <- sample(motifs, 50)
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(pfms)
hc <- MotIV::motifHclust(d)
phylog <- hclust2phylog(hc)
leaves <- names(phylog$leaves)
pfms <- pfms[leaves]
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm",mat=pfms[[.ele]], name=.ele)},pfms)
pfms <- DNAmotifAlignment(pfms, minimalConsensus=3)
library(RColorBrewer)
color <- brewer.pal(12, "Set3")
plotMotifStackWithRadialPhylog(phylog, pfms, circle=0.9, cleaves = 0.5, clabel.leaves = 0.7,
 col.bg=rep(color, each=5), col.leaves=rep(color, each=5))
}
```

---

plotXaxis *plot x-axis*

Description

plot x-axis for the sequence logo

Usage

```
plotXaxis(pfm, p=rep(0.25, 4))
```

Arguments

pfm	position frequency matrices
p	background possibility

Value

none

plotYaxis *plot y-axis*

Description

plot y-axis for the sequence logo

Usage

```
plotYaxis(pfm)
```

Arguments

pfm	position frequency matrices

Value

none

readPCM *read pcm from a path*

Description

read position count matrix from a path

Usage

```
readPCM(path=".", pattern=NULL)
```

Arguments

path	a character vector of full path names
pattern	an optional regular expression

Value

A list of position count matrix

Examples

```
pcms<-readPCM(file.path(find.package("motifStack"), "extdata"),"pcm$")
```

Index