Package ‘motifStack’

March 23, 2017

Type Package
Version 1.19.2
Date 2017-02-10
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 XML, scales, htmlwidgets
Depends R (>= 2.15.1), methods, grImport, grid, MotIV, ade4, Biostrings
Suggests RUnit, BiocGenerics, MotifDb, RColorBrewer, BiocStyle, knitr
biocViews SequenceMatching, Visualization, Sequencing, Microarray, Alignment, ChIPchip, ChIPSeq, MotifAnnotation, DataImport
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
VignetteBuilder knitr
NeedsCompilation no

R topics documented:

motifStack-package .................................................. 2
browseMotifs .......................................................... 3
browseMotifs-shiny ..................................................... 4
colorset ................................................................. 4
DNAmotifAlignment .................................................... 5
getRankedUniqueMotifs ............................................... 5
highlightCol ............................................................ 6
mergeMotifs ............................................................. 7
motifCircos ............................................................. 8
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>
browseMotifs

**Description**

browse motifs in a web browser

**Usage**

```r
browseMotifs(pfms, phylog,
layout=c("tree", "cluster", "radialPhylog"),
nodeRadius=2.5, baseWidth=12, baseHeight=30,
xaxis=TRUE, yaxis=TRUE,
width=NULL, height=NULL,
...
```

**Arguments**

- `pfms` a list of `pfm`
- `phylog` layout type. see `GraphvizLayouts`
- `layout` layout type. Could be tree, cluster or radialPhylog.
- `nodeRadius` node radius, default 2.5px.
- `baseWidth`, `baseHeight` width and height of each alphabet of the motif logo.
- `xaxis`, `yaxis` plot x-axis or y-axis or not in the motifs.
- `width` width of the figure
- `height` height of the figure
- `...` parameters not used

**Value**

An object of class htmlwidget that will intelligently print itself into HTML in a variety of contexts including the R console, within R Markdown documents, and within Shiny output bindings.

**Examples**

```r
library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
motifs <- as.list(matrix.fly)
motifs <- motifs[grepl("[Dd]melanogaster-[Ff]lyFactorSurvey-", 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, 10)
```

```r
pfms <- lapply(names(pfms), function(.ele, pfms){new("pfm", mat=pfms[[.ele]], name=.ele)}, pfms)
browseMotifs(pfms)
```
**browseMotifs-shiny**  
*Shiny bindings for browseMotifs*

**Description**

Output and render functions for using browseMotifs within Shiny applications and interactive Rmd documents.

**Usage**

```r
browseMotifsOutput(outputId, width = "100\%", height = "400px")
renderbrowseMotifs(expr, env = parent.frame(), quoted = FALSE)
```

**Arguments**

- `outputId` output variable to read from
- `width, height` Must be a valid CSS unit (like '100\%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
- `expr` An expression that generates a browseMotifs
- `env` The environment in which to evaluate expr.
- `quoted` Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

**colorset**  
*retrieve color setting for logo*

**Description**

retrieve color setting for logo

**Usage**

```r
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**

```r
col <- colorset("AA", "hydrophobicity")
```
**Description**

align DNA motifs for plotting motifs stack

**Usage**

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

**Arguments**

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

**Value**

a list of aligned motifs

**Examples**

```
pcms<-readPCM(file.path(find.package("motifStack"), "extdata"),"pcm$")
motifs<-lapply(pcms,pcm2pfm)
motifs<-DNAmotifAlignment(motifs)
```

---

**getRankedUniqueMotifs**

get the unique motif in each category grouped by distance

**Description**

to get the unique motif in a given category, eg by species.

**Usage**

```
getRankedUniqueMotifs(phylog, attr)
```

**Arguments**

- **phylog**: an object of class phylog
- **attr**: attribute used for category of motifs
Value

return a list:

uni.rank  unique motif ranks
uni.length length of unique motif grouped by distance
uni.list  unique motif names grouped by distance

Author(s)

Jianhong Ou

Examples

if(interactive()){
  library("MotifDb")
  matrix.fly <- query(MotifDb, "Dmelanogaster")
  matrix.human <- query(MotifDb, "Hsapiens")
  pfms <- c(as.list(matrix.fly), as.list(matrix.human))
  pfms <- pfms[sample(1:length(pfms), 100)]
  jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
                                          "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfms, pfm2pwm))
  hc <- MotIV::motifHclust(d, method="average")
  phylog <- hclust2phylog(hc)
  leaves <- names(phylog$leaves)
  attr <- gsub("^(.*?)_.*$", "\1", leaves)
  getRankedUniqueMotifs(phylog, attr)
}

highlightCol

add alpha transparency value to a color

Description

An alpha transparency value can be specified to a color, in order to get better color for background.

Usage

highlightCol(col, alpha = 0.5)

Arguments

col  vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].

alpha a value in [0, 1]

Value

a vector of colors in hexadecimal string of the form "#rrggbbaa".
mergeMotifs

Author(s)
Jianhong Ou

Examples
highlightCol(1:5, 0.3)
highlightCol(c("red", "green", "blue"), 0.3)

mergeMotifs
merge multiple motifs

Description
merge multiple motifs by calculate mean of each position

Usage
mergeMotifs(..., bgNoise=NA)

Arguments
... pcm or pfm objects
bgNoise if it is not NA, test will using a background by Dirichlet(1)-distributed random frequencies with weight bg.noise. The value of bgNoise should be a number in the range of 0 to 1, eg. 0.05

Value
a pfm object

Author(s)
Jianhong Ou

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

plot sequence logo stacks with a radial phylogenetic tree and multiple color rings.

Description

plot sequence logo stacks with a radial phylogenetic tree and multiple color rings. The difference from plotMotifStackWithRadialPhylog is that it has more color setting and one more group of pfms.

Usage

motifCircos(phylog, pfms=NULL, pfms2=NULL, R=2.5,
  r.tree=1, col.tree.bg=NULL, col.tree.bg.alpha=1,
  cnodes=0, labels.nodes=names(phylog$nodes), clabel.nodes=0,
  r.leaves=NA,
  cleaves=1, labels.leaves=names(phylog$leaves), clabel.leaves=1,
  col.leaves=rep("black", length(labels.leaves)),
  col.leaves.bg=NULL, col.leaves.bg.alpha=1,
  r.pfms=NA, r.pfms2=NA,
  r.rings=0, col.rings=list(),
  col.inner.label.circle=NULL, inner.label.circle.width=0.02,
  col.outer.label.circle=NULL, outer.label.circle.width=0.02,
  draw.box=FALSE,
  clockwise =FALSE, init.angle=if(clockwise) 90 else 0,
  angle=360, pfmNameSpliter=";", rcpostfix="(RC)",
  motifScale="linear","logarithmic"), ic.scale=TRUE,
  plotIndex=FALSE, IndexCol="black", IndexCex=.8,
  groupDistance=NA, groupDistanceLineCol="red",
  plotAxis=FALSE)

Arguments

phylog  an object of class phylog
pfms  a list of objects of class pfm
pfms2  a list of objects of class pfm
R  radius of canvas
r.tree  half width of the tree
col.tree.bg  a vector of colors for tree background
col.tree.bg.alpha  a alpha value [0, 1] of colors for tree background
cnodes  a character size for plotting the points that represent the nodes, used with par("cex")*cnodes. If zero, no points are drawn
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
r.leaves  width of the leaves
cleaves  a character size for plotting the points that represent the leaves, used with par("cex")*cleaves. 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
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 background of leaves labels
r.pfms  width of the pfms
r.pfms2  width of the pfms2
r.rings  a vector of width of color rings
col.rings  a list of color rings
col.inner.label.circle  
a vector of colors for inner circle of pfms
inner.label.circle.width  
width for inner circle of pfms
col.outer.label.circle  
a vector of colors for outer circle of pfms
outer.label.circle.width  
width for outer circle of pfms
draw.box  if TRUE draws a box around the current plot with the function box()
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 phylogenetic tree. Defaults 360
pfmNameSpliter  splitter when name of pfms/pfms2 contain multiple node of labels.leaves
rcpostfix  the postfix for reverse complements
motifScale  the scale of logo size
ic.scale  logical. If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
plotIndex  logical. If TRUE, will plot index number in the motifLogo which can help user to describe the motifLogo
IndexCol  The color of the index number when plotIndex is TRUE.
IndexCex  The cex of the index number when plotIndex is TRUE.
groupDistance  show groupDistance on the draw
groupDistanceLineCol  groupDistance line color, default: red
plotAxis  logical. If TRUE, will plot distance axis.

Value

none

Author(s)

Jianhong Ou
motifCloud

See Also
plotMotifStackWithRadialPhylog

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]+]+$_", "
  names(motifs)))
  motifs <- motifs[unique(names(motifs))]
  pfm.s <- sample(motifs, 50)
  jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
  "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfm.s, pfm2pwm))
  hc <- MotIV::motifHclust(d, method="average")
  phylog <- hclust2phylog(hc)
  leaves <- names(phylog$leaves)
  pfm.s <- lapply(names(pfm.s), function(.ele, pfm.s){new("pfm", mat=pfm.s[[.ele]],
    name=.ele)}, pfm.s)
  pfm.s <- DNAmotifAlignment(pfm.s, minimalConsensus=3)
  library(RColorBrewer)
  color <- brewer.pal(12, "Set3")
  motifCircos(phylog, pfm.s, cleaves = 0.5, clabel.leaves = 0.7,
    col.tree.bg=rep(color, each=5), col.leaves=rep(color, each=5),
    r.rings=c(0.02, 0.03, 0.04),
    col.rings=list(sample(colors(), 50),
      sample(colors(), 50),
      sample(colors(), 50)))
}

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,
  font="Helvetica-Bold", ic.scale=TRUE, fontsize=12)
motifCloud

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
- **font**: font of logo
- **ic.scale**: logical If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
- **fontsize**: font size of the template for grImport, default 12. Higher value make better quality figure, but also increase the file size.

Value

none

Examples

```r
if(interactive()){
  library("MotifDb")
  matrix.fly <- query(MotifDb, "Dmelanogaster")
  motifs <- as.list(matrix.fly)
  motifs <- motifs[grep("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
  names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", "",
                        gsub("_FBgn[0-9]+$", "",
                        gsub("[^a-zA-Z0-9]","_",
                        gsub("(_[0-9]+)+$", "",
                        gsub("([^0-9])+$", "", names(motifs)))))
  motifs <- motifs[unique(names(motifs))]
  pfms <- sample(motifs, 50)
  jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
                        "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfms, pfm2pwm))
  hc <- MotIV::motifHclust(d, method="average")
  phylolg <- hclust2phylog(hc)
  leaves <- names(phylolg$leaves)
  pfms <- lapply(leaves, function(.ele, pfms){new("pfm", mat=pfms[[.ele]],
                        name=.ele)})
  motifSig <- motifSignature(pfms, phylolg, groupDistance=0.1)
  motifCloud(motifSig)
}
```
motifPiles  

plot sequence logo stacks with a linear phylogenic tree and multiple color sets

Description

plot sequence logo stacks with a linear phylogenic tree and multiple color sets.

Usage

```r
motifPiles(phylog, pfms=NULL, pfms2=NULL,
           r.tree=.45, col.tree=NULL,
           cnodes=0, labels.nodes=names(phylog$nodes), clabel.nodes=0,
           cleaves=.2, labels.leaves=names(phylog$leaves), clabel.leaves=1,
           col.leaves=rep("black", length(labels.leaves)),
           col.leaves.bg=NULL, col.leaves.bg.alpha=1,
           r.pfms=NA, r.pfms2=NA, motifScale=c("logarithmic", "linear"),
           col.pfms=NULL, col.pfms.width=0.02,
           col.pfms2=NULL, col.pfms2.width=0.02,
           r.anno=0, col.anno=list(),
           pfmNameSpliter=";", rcpostfix="(RC)", ic.scale=TRUE,
           plotIndex=FALSE, IndexCol="black", IndexCex=.8,
           groupDistance=NA, groupDistanceLineCol="red")
```

Arguments

- `phylog`  
an object of class phylog
- `pfms`  
a list of objects of class pfm
- `pfms2`  
a list of objects of class pfm
- `r.tree`  
width of the tree
- `col.tree`  
a vector of colors for tree
- `cnodes`  
a character size for plotting the points that represent the nodes, used with `par("cex")*cnodes`. If zero, no points are drawn
- `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
- `cleaves`  
a character size for plotting the points that represent the leaves, used with `par("cex")*cleaves`. 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
- `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 background of leaves labels
- `r.pfms`  
width of the pfms
- `r.pfms2`  
width of the pfms2
motifPiles

motifScale the scale of logo size
col.pfms a vector of colors for inner pile of pfms
col.pfms.width width for inner pile of pfms
col.pfms2 a vector of colors for outer pile of pfms
col.pfms2.width width for outer pile of pfms
r.anno a vector of width of color sets
col.anno a list of color sets
pfmNameSpliter spliter when name of pfms/pfms2 contain multiple node of labels.leaves
rcpostfix the postfix for reverse complements
ic.scale logical. If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
plotIndex logical. If TRUE, will plot index number in the motifLogo which can help user to describe the motifLogo
IndexCol The color of the index number when plotIndex is TRUE.
IndexCex The cex of the index number when plotIndex is TRUE.
groupDistance show groupDistance on the draw
groupDistanceLineCol groupDistance line color, default: red

Value

none

Author(s)

Jianhong Ou

See Also

motifCircos

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]+$","",
                           gsub("[a-zA-Z-0-9]","_",
                                 gsub("(_[0-9]+)+$","",
                                      names(motifs)))))
  motifs <- motifs[unique(names(motifs))]
  pfms <- sample(motifs, 50)
  jasper.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
                                         "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfms, pfm2pwm))
  hc <- MotIV::motifHclust(d, method="average")
  phylog <- hclust2phylog(hc)
  leaves <- names(phylog$leaves)
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, gpcol).

Slots

- signatures: list object of class "pfm"
- freq: code "numeric" signature frequency
- nodelist: list object of class "ouNode"
- gpcol: code "character" signature group color sets

Methods

- signatures: signature(object = "motifSig") return the signatures of motifSig
- frequency: signature(object = "motifSig") return the frequency of motifSig
- nodelist: signature(object = "motifSig") return the nodelist of motifSig
- sigColor: signature(object = "motifSig") return the group color sets of motifSig

```r
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")
motifPiles(phylog, pfms, cleaves = 0.5, clabel.leaves = 0.7,
  col.leaves=rep(color, each=5),
  col.leaves.bg = sample(colors(), 50),
  col.tree=rep(color, each=5),
  r.anno=c(0.02, 0.03, 0.04),
  col.anno=list(sample(colors(), 50),
                sample(colors(), 50)))
```
Description

methods for motifSig objects.

Usage

```r
## S4 method for signature 'motifSig'
signatures(object)
## S4 method for signature 'motifSig'
frequency(object)
## S4 method for signature 'motifSig'
nodelist(object)
## S4 method for signature 'motifSig'
sigColor(object)
```

Arguments

- `object`: An object of class motifSig.

Methods

- `signatures`: signature(object = "motifSig") return the signatures of motifSig
- `frequency`: signature(object = "motifSig") return the frequency of motifSig
- `nodelist`: signature(object = "motifSig") return the nodelist of motifSig
- `sigColor`: signature(object = "motifSig") return the group color sets of motifSig
- `$, $<-`: Get or set the slot of motifSig

Description

extract signatures from multiple motifs by distance calculated from STAMP

Usage

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

plot a DNA sequence logo stack

Description

Plot a DNA sequence logo stack

Usage

motifStack(pfms, layout=c("stack", "treeview", "phylog", "radialPhylog"), ...)
ouNode-class

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

- return a list contains pfms and phylog

Examples

```r
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]+$", "",
                             gsub("([a-zA-Z0-9]+)\-", "",
                                  gsub("([a-zA-Z0-9]+)\+$", "", 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

Methods

- **$**, **$<-**: Get or set the slot of `ouNode`
Examples

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

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 = "pcm", 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_SOXEXA.pcm"))
pcm <- pcm[,3:ncol(pcm)]
rownames(pcm) <- c("A","C","G","T")
motif <- new("pcm", mat=as.matrix(pcm), name="bin_SOXEXA")
plot(motif)
Description

methods for pcm objects.

Usage

```r
## 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 `pcm`. 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 = "pcm", t= "numeric") trim motif by information content.

$, $<- Get or set the slot of pcm

Examples

```r
cmc <- read.table(file.path(find.package("motifStack"), "extdata", "bin_SOLEXA.pcm"))
cmc <- cmc[,3:ncol(cmc)]
rownames(cmc) <- c("A","C","G","T")
motif <- new("pcm", mat=as.matrix(cmc), 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"character" vector. The color setting for each symbol
- `background"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)

Description

methods for pfm objects.

Usage

## S4 method for signature 'pfm,numerical,logical'
addBlank(x,n,b)
## S4 method for signature 'pfm,ANY'
getIC(x,p="missing")
## S4 method for signature 'matrix,numerical'
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,numerical'
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.

$\$, $<- Get or set the slot of pfm

Examples

```r
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")
```

---

**pfm2_pwm**

*convert pfm object to PWM*

Description

convert pfm object to PWM

Usage

`pfm2_pwm(x)`

Arguments

- `x`: an object of pfm or pcm or matrix

Value

A numeric matrix representing the Position Weight Matrix for PWM.

Author(s)

Jianhong Ou
plotMotifLogo

See Also

PWM

Examples

library("MotifDb")
matrix.fly <- query(MotifDb, "Dmelanogaster")
pfm2pwm(matrix.fly[[1]])

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, ic.scale=TRUE, fontsize=12)

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
  ic.scale  logical If TRUE, the height of each column is proportional to its information
            content. Otherwise, all columns have the same height.
  fontsize  font size of the template for grImport, default 12. Higher value make better
            quality figure, but also increase the file size.

Value

  none
Examples

```r
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

```r
plotMotifLogoA(pfm, font="Helvetica-Bold", ic.scale=TRUE, fontsize=12)
```

Arguments

- `pfm`: an object of pfm
- `font`: font of logo
- `ic.scale`: logical, If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
- `fontsize`: font size of the template for grImport, default 12. Higher value make better quality figure, but also increase the file size.

Value

none

Examples

```r
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

```r
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

**Examples**

```r
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)
plotMotifLogoStackWithTree(pfms)
```
plotMotifOverMotif

Description

plot motif over another motif to emphasize the difference.

Usage

plotMotifOverMotif(motif, backgroundMotif, bgNoise=NA,
                    font="Helvetica-Bold", textgp=gpar())

Arguments

motif an object of pcm or pfm
backgroundMotif an object of pcm or pfm
bgNoise if it is not NA, test will using a background by Dirichlet(1)-distributed random
        frequencies with weight bg.noise. The value of bgNoise should be a number in
        the range of 0 to 1, eg. 0.05
font font for logo symbol
textgp text parameter

Value

none
plotMotifStackWithPhylog

Examples

```r
pcms <- readPCM(file.path(find.package("motifStack"), "extdata"), "pcm")
len <- sapply(pcms, function(.ele) ncol(.ele$mat))
pcms <- pcms[len==7]
plotMotifOverMotif(pcms[[1]], pcms[[2]], bgNoise=0.05)
```

Description

plot sequence logo stacks with a ape4-style phylogenic tree

Usage

```r
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,
  font="Helvetica-Bold", ic.scale=TRUE, fontsize=12)
```

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
- `font`: font of logo
- `ic.scale`: logical If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
- `fontsize`: font size of the template for grImport, default 12. Higher value make better quality figure, but also increase the file size.

Value

none
**plotMotifStackWithRadialPhylog**

**plot sequence logo stacks with a radial phylogenic tree**

**Description**

plot sequence logo stacks with a radial phylogenic tree

**Usage**

```r
plotMotifStackWithRadialPhylog(phylog, pfms==NULL, 
  circle=0.75, 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, inner.label.circle.width="default", 
  col.outer.label.circle=NULL, outer.label.circle.width="default", 
  clockwise =FALSE, init.angle=if(clockwise) 90 else 0, 
  angle=360, pfmNameSpliter=";", rcpostfix = "(RC)", 
  motifScale=c("linear","logarithmic"), ic.scale=TRUE, 
  ...)
```

**Examples**

```r
if(interactive()){ 
  library("MotifDb")
  matrix.fly <- query(MotifDb, "Dmelanogaster")
  motifs <- as.list(matrix.fly)
  motifs <- motifs[grep("Dmelanogaster-FlyFactorSurvey-", names(motifs), fixed=TRUE)]
  names(motifs) <- gsub("Dmelanogaster_FlyFactorSurvey_", ", 
  gsub("_FBgn\[0-9]+$", "", 
  gsub("[^a-zA-Z0-9]+","", names(motifs))))
  motifs <- motifs[unique(names(motifs))]
  pfms <- sample(motifs, 50)
  jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"), 
    "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfms, pfm2pwm))
  hc <- MotIV::motifHclust(d, method="average")
  phylog <- hclust2phylog(hc)
  leaves <- names(phylog$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)}
```

**See Also**

plot.phylog
plotMotifStackWithRadialPhylog

plotIndex=FALSE, IndexCol="black", IndexCex=.8,
groupDistance=NA, groupDistanceLineCol="red",
plotAxis=FALSE, font="Helvetica-Bold", fontsize=12)

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
glabels.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 background of leaves labels
col.bg a vector of colors for tree background
col.bg.alpha a alpha value [0, 1] of colors for tree background
col.inner.label.circle a vector of colors for inner circle of pfms
inner.label.circle.width width for inner circle of pfms
col.outer.label.circle a vector of colors for outer circle of pfms
outer.label.circle.width width for outer circle of pfms
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
pfmNameSplitter spliter when name of pfms contain multiple node of labels.leaves
rcpostfix the postfix for reverse complements
motifScale the scale of logo size
ic.scale logical. If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.
plotMotifStackWithRadialPhylog

plotIndex logical. If TRUE, will plot index number in the motifLogo which can help user to describe the motifLogo

IndexCol The color of the index number when plotIndex is TRUE.

IndexCex The cex of the index number when plotIndex is TRUE.

groupDistance show groupDistance on the draw

groupDistanceLineCol groupDistance line color, default: red

plotAxis logical. If TRUE, will plot distance axis.

font font of logo

fontsize font size of the template for grImport, default 12. Higher value make better quality figure, but also increase the file size.

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, method="average")  
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**

---

**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**

---

**Description**

plot y-axis for the sequence logo

**Usage**

plotYaxis(ymax)

**Arguments**

- `ymax` max value of y axis

**Value**

none
readPCM

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 pcm objects

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

reorderUPGMAtree

Description
re-order the UPGMA tree by adjacent motif distance

Usage
reorderUPGMAtree(phylog, motifs, rcpostfix = "(RC)")

Arguments
phylog an object of phylog
motifs a list of objects of pfm
rcpostfix the postfix for reverse complements

Value
an object of phylog

Author(s)
Jianhong Ou
Examples

```r
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("_[FBgn0-9]+$", "",
      gsub("[a-zA-Z0-9-]_","",
        gsub("([0-9]+)+$", "", 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, method="average")
  phylog <- hclust2phylog(hc)
  reorderUPGMAtree(phylog, pfms)
}
```
Index

*Topic classes
- motifSig-class, 14
- motifSig-methods, 15
- ouNode-class, 17
- pcm-class, 18
- pcm-methods, 19
- pfm-class, 20
- pfm-methods, 21

*Topic misc
- getRankedUniqueMotifs, 5
- highlightCol, 6
- mergeMotifs, 7
- motifCircos, 8
- motifPiles, 12
- pfm2pwm, 22
- reorderUPGMAtree, 32

*Topic package
- motifStack-package, 2

*Topic plot
- browseMotifs, 3
- $, motifSig-method (motifSig-methods), 15
- $, ouNode-method (ouNode-class), 17
- $, pcm-method (pcm-methods), 19
- $, pfm-method (pfm-methods), 21
- $<-, motifSig-method (motifSig-methods), 15
- $<-, ouNode-method (ouNode-class), 17
- $<-, pcm-method (pcm-methods), 19
- $<-, pfm-method (pfm-methods), 21
- addBlank (pfm-methods), 21
- addBlank, pcm, numeric, logical-method (pcm-methods), 19
- addBlank, pfm, numeric, logical-method (pfm-methods), 21
- browseMotifs, 3
- browseMotifs-shiny, 4
- browseMotifsOutput
  (browseMotifs-shiny), 4
- colors, 6
- colorset, 4
- DNAmotifAlignment, 5
- frequence (motifSig-methods), 15
- frequence, motifSig-method (motifSig-methods), 15
- getIC (pfm-methods), 21
- getIC, matrix, matrix-method (pfm-methods), 21
- getIC, matrix, numeric-method (pfm-methods), 21
- getIC, pcm, ANY-method (pcm-methods), 19
- getIC, pfm, ANY-method (pfm-methods), 21
- getRankedUniqueMotifs, 5
- GraphvizLayouts, 3
- highlightCol, 6
- matrixReverseComplement (pfm-methods), 21
- matrixReverseComplement, pcm-method (pcm-methods), 19
- matrixReverseComplement, pfm-method (pfm-methods), 21
- mergeMotifs, 7
- motifCircos, 8, 13
- motifCloud, 10
- motifPiles, 12
- motifSig, 11, 15, 16
- motifSig (motifSig-methods), 15
- motifSig-class, 14
- motifSig-methods, 15
- motifSignature, 14, 15
- motifStack, 16
- motifStack-package, 2
- nodelist (motifSig-methods), 15
- nodelist, motifSig-method (motifSig-methods), 15
- ouNode, 14, 17
- ouNode (ouNode-class), 17
- ouNode-class, 17
- palette, 6
- pcm, 7, 20, 22, 26, 32
- pcm (pcm-methods), 19
INDEX

pcm-class, 18
pcm-methods, 19
pcm2pfm (pcm-methods), 19
pcm2pfm, data.frame, ANY-method
  (pcm-methods), 19
pcm2pfm, data.frame, numeric-method
  (pcm-methods), 19
pcm2pfm, matrix, ANY-method
  (pcm-methods), 19
pcm2pfm, matrix, numeric-method
  (pcm-methods), 19
pcm2pfm, pcm, ANY-method (pcm-methods), 19
pfm, 3, 7, 17, 22, 26
pfm (pfm-methods), 21
pfm-class, 20
pfm-methods, 21
pfm2pwm, 22
plot, pcm, ANY-method (pcm-methods), 19
plot, pfm, ANY-method (pfm-methods), 21
plot.phylog, 28, 30
plotMotifLogo, 23
plotMotifLogoA, 24
plotMotifLogoStack, 17, 24
plotMotifLogoStackWithTree, 17, 25
plotMotifOverMotif, 26
plotMotifStackWithPhylog, 17, 27
plotMotifStackWithRadialPhylog, 10, 17, 28
plotXaxis, 31
plotYaxis, 31
PWM, 23
readPCM, 32
renderbrowseMotifs
  (browseMotifs-shiny), 4
reorderUPGMAtree, 32
rgb, 6
sigColor (motifSig-methods), 15
sigColor, motifSig-method
  (motifSig-methods), 15
signatures (motifSig-methods), 15
signatures, motifSig-method
  (motifSig-methods), 15
trimMotif (pcm-methods), 19
trimMotif, pcm, numeric-method
  (pcm-methods), 19
trimMotif, pfm, numeric-method
  (pfm-methods), 21