Package ‘motifStack’

January 30, 2017

Type Package

Version 1.18.0

Date 2015-10-9

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

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
colorset ................................................................. 2
DNAmotifAlignment .................................................. 3
getRankedUniqueMotifs .............................................. 4
highlightCol ............................................................ 5
mergeMotifs ............................................................. 5
motifCircos ............................................................. 6
motifCloud ............................................................... 8
motifPiles ............................................................... 10
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')
**DNAmotifAlignment**

Arguments

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

Value

A character vector of color scheme

Examples

```
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 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$")
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**

```r
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", or a positive integer meaning `palette()`. 
- `alpha`: a value in [0, 1]

**Value**

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

**Author(s)**

Jianhong Ou

**Examples**

```r
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**

```r
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
motifCircos

Value

a pfm object

Author(s)

Jianhong Ou

Examples

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

```
# plot sequence logo stacks with a radial phylogenic tree and multiple color rings

motifCircos

Description

plot sequence logo stacks with a radial phylogenic 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, pfmNameSplitter=";", rcpostfix="(RC)",
             motifScale=c("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
motifCircos

- `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 backgroud 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 cirlce 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 phylogenic tree. Defaults 360
- `pfmNameSplitter` - spliter 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.
motifCloud

Plot a DNA sequence logo cloud

### Description

Plot a DNA sequence logo cloud
motifCloud

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)

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

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]+$", ",",
      gsub("[a-zA-Z0-9]+","","
        gsub("(_[0-9]+)+","","
          names(motifs)))))
  motifs <- motifs[unique(names(motifs))]
  pfms <- sample(motifs, 50)
  jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
    "extdata", "jaspar2010_PCC_SWU.scores"))
  d <- MotIV::motifDistances(lapply(pfms, pfm2pwm))
motifPiles <- MotIV::motifHclust(d, method="average")
phylog <- hclust2phylog(hc)
leaves <- names(phylog$leaves)
pfms <- pfms[leaves]
  pfms <- lapply(names(pfms), function(.ele, pfms)(new("pfm", mat=pfms[[.ele]],
name=.ele)),pfms)
motifSig <- motifSignature(pfms, phylog, 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
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.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,
  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
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
pfmNameSplitter: splitter 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]++", "",
})
```r
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(lapply(pfms, pfm2pwm))

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")
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),
           sample(colors(), 50)))
```

---

**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
motifSig-methods

Description

methods for motifSig objects.

Usage

## 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

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())
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

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

Methods

$, $<- Get or set the slot of ouNode
**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 = "pcm", t= "numeric") trim motif by information content.
- **plot** signature(x = "pcm") Plots the sequence logo of the position count matrix.

**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 <- new("pcm", mat=as.matrix(pcm), name="bin_SOLEXA")
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
pfm-class

Get or set the slot of pcm

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)

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.
pfm-methods

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,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.

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

Examples

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

---

**pfm2PWM** convert pfm object to PWM

Description

convert pfm object to PWM

Usage

```r
pfm2PWM(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",
xlce=1.2, ylce=1.2, ncex=1.2, ic.scale=TRUE, fontsize=12)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pfm</td>
<td>a position frequency matrices</td>
</tr>
<tr>
<td>motifName</td>
<td>motif name</td>
</tr>
<tr>
<td>p</td>
<td>background possibility</td>
</tr>
<tr>
<td>font</td>
<td>font of logo</td>
</tr>
<tr>
<td>colset</td>
<td>color setting for each logo letter</td>
</tr>
<tr>
<td>xaxis</td>
<td>draw x-axis or not</td>
</tr>
<tr>
<td>yaxis</td>
<td>draw y-axis or not</td>
</tr>
<tr>
<td>xlab</td>
<td>x-label, do nothing if set xlab as NA</td>
</tr>
<tr>
<td>ylab</td>
<td>y-label, do nothing if set ylab as NA</td>
</tr>
<tr>
<td>xlce</td>
<td>cex value for x-label</td>
</tr>
<tr>
<td>ylce</td>
<td>cex value for y-label</td>
</tr>
<tr>
<td>ncex</td>
<td>cex value for motif name</td>
</tr>
<tr>
<td>ic.scale</td>
<td>logical If TRUE, the height of each column is proportional to its information content. Otherwise, all columns have the same height.</td>
</tr>
<tr>
<td>fontsize</td>
<td>font size of the template for grImport, default 12. Higher value make better quality figure, but also increase the file size.</td>
</tr>
</tbody>
</table>

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_SOEXA")
plotMotifLogoA(motif)
```

---

**plotMotifLogoStack**  
plot sequence logos stack

Description

plot sequence logos stack

Usage

```r
plotMotifLogoStack(pfms, ...)
```
plotMotifLogoStackWithTree

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
plotMotifOverMotif

Value
none

Examples

#####Input#####
pcms<-readPCM(file.path(find.package("motifStack"), "extdata","pcm"))
motifs<-lapply(pcms,pcm2pfm)

#####Clustering#####
jaspar.scores <- MotIV::readDBScores(file.path(find.package("MotIV"),
                                      "extdata", "jaspar2010_PCC_SWU.scores"))
d <- MotIV::motifDistances(lapply(motifs, pfm2pwm))
hc <- MotIV::motifHclust(d, method="average")

##reorder the motifs for plotMotifLogoStack
motifs<-motifs[hc$order]
##do alignment
motifs<-DNAmotifAlignment(motifs)
##plot stacks
plotMotifLogoStack(motifs, ncex=1.0)
plotMotifLogoStackWithTree(motifs, hc=hc)

Description
plot motif over another motif to emphesize 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)
```

---

**plotMotifStackWithPhylog**

> plot sequence logo stacks with a ape4-style phylogenic tree

**Description**

plot sequence logo stacks with a ape4-style phylogenetic 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 phylogenetic tree

Description

plot sequence logo stacks with a radial phylogenetic tree

Usage

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=TRUE, inner.label.circle.width="default",
col.outer.label.circle=TRUE, 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,
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 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 splitter 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 <- lapply(leaves, 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**

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

**Description**

plot x-axis for the sequence logo

**Arguments**

- `pfm` position frequency matrices
- `p` background possibility

**Value**

none

**plotYaxis**

```
plotYaxis(ymax)
```

**Description**

plot y-axis for the sequence logo

**Usage**

```
plotYaxis(ymax)
```

**Arguments**

- `ymax` max value of y axis

**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 `pcm` objects

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

---

**reorderUPGMAtree**  
*re-order UPGMA tree*

**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("_FBgn[0-9]+$", ", ",
                            gsub("[^a-zA-Z0-9]+", ", ", 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(pfms)
  hc <- MotIV::motifHclust(d, method="average")
  phylog <- hclust2phylog(hc)
  reorderUPGMAtree(phylog, pfms)
}
```
Index

*Topic* classes
motifSig-class, 12
motifSig-methods, 13
ouNode-class, 15
pcm-class, 16
pcm-methods, 17
pfm-class, 18
pfm-methods, 19

*Topic* misc
getRankedUniqueMotifs, 4
highlightCol, 5
mergeMotifs, 5
motifCircos, 6
motifPiles, 10
pfm2pfm, 20
reorderUPGMAtree, 30

*Topic* package
motifStack-package, 2
$.motifSig-method (motifSig-methods), 13
$.ouNode-method (ouNode-class), 15
$.pcm-method (pcm-methods), 17
$.pfm-method (pfm-methods), 19
$<-,motifSig-method (motifSig-methods), 13
$<-,ouNode-method (ouNode-class), 15
$<-,pcm-method (pcm-methods), 17
$<-,pfm-method (pfm-methods), 19
addBlank (pfm-methods), 19
addBlank,pcm,numERIC,logical-method (pcm-methods), 17
addBlank,pfm,numERIC,logical-method (pfm-methods), 19

colors, 5
colorset, 2
DNAmotifAlignment, 3
frequency (motifSig-methods), 13
frequency,motifSig-method (motifSig-methods), 13
getIC (pfm-methods), 19
getIC,matrix,matrix-method (pfm-methods), 19
getIC,matrix,numERIC-method (pfm-methods), 19
getIC,pcm,ANY-method (pcm-methods), 17
getIC,pfm,ANY-method (pfm-methods), 19
getRankedUniqueMotifs, 4
highlightCol, 5
matrixReverseComplement (pfm-methods), 19
matrixReverseComplement,pcm-method (pcm-methods), 17
matrixReverseComplement,pfm-method (pfm-methods), 19
mergeMotifs, 5
motifCircos, 6, 11
motifCloud, 8
motifPiles, 10
motifSig, 9, 13, 14
motifSig (motifSig-methods), 13
motifSig-class, 12
motifSig-methods, 13
motifSignature, 12, 13
motifStack, 14
motifStack-package, 2
nodelist (motifSig-methods), 13
nodelist,motifSig-method (motifSig-methods), 13
ouNode, 12, 15
ouNode (ouNode-class), 15
ouNode-class, 15
palette, 5
pcm, 5, 18, 20, 24, 30
pcm (pcm-methods), 17
pcm-class, 16
pcm-methods, 17
pcm2pfm (pcm-methods), 17
pcm2pfm,data.frame,ANY-method (pcm-methods), 17

32
INDEX

pcm2pfm, data.frame, numeric-method
  (pcm-methods), 17
pcm2pfm, matrix, ANY-method
  (pcm-methods), 17
pcm2pfm, matrix, numeric-method
  (pcm-methods), 17
pcm2pfm, pcm, ANY-method (pcm-methods), 17
pfm, 5, 6, 15, 20, 24
pfm-methods, 19
pfm-class, 18
pfm-methods, 19
pfm2pwm, 20
plot, pcm, ANY-method (pcm-methods), 17
plot, pfm, ANY-method (pfm-methods), 19
plot.phylog, 26, 28
plotMotifLogo, 21
plotMotifLogoA, 22
plotMotifLogoStack, 15, 22
plotMotifLogoStackWithTree, 15, 23
plotMotifOverMotif, 24
plotMotifStackWithPhylog, 15, 25
plotMotifStackWithRadialPhylog, 8, 15, 26
plotXaxis, 29
plotYaxis, 29
PWMM, 21

readPCM, 30
reorderUPGMAtree, 30
rgb, 5

sigColor (motifSig-methods), 13
sigColor, motifSig-method
  (motifSig-methods), 13
signatures (motifSig-methods), 13
signatures, motifSig-method
  (motifSig-methods), 13

trimMotif (pcm-methods), 17
trimMotif, pcm, numeric-method
  (pcm-methods), 17
trimMotif, pfm, numeric-method
  (pfm-methods), 19