Package `dagLogo`

April 14, 2017

**Type** Package  
**Title** dagLogo  
**Version** 1.12.0  
**Date** 2016-05-03  
**Author** Jianhong Ou, Alexey Stukalov, Niraj Nirala, Usha Acharya, Lihua Julie Zhu  
**Maintainer** Jianhong Ou <jianhong.ou@umassmed.edu>  
**Description**  
Visualize significant conserved amino acid sequence pattern in groups based on probability theory.  
**License** GPL (>=2)  
**Depends** R (>= 3.0.1), methods, biomaRt, grImport, grid, motifStack  
**Imports** pheatmap, Biostrings  
**Suggests** XML, UniProt.ws, BiocStyle, knitr, rmarkdown, testthat  
**biocViews** SequenceMatching, Visualization  
**VignetteBuilder** knitr  
**NeedsCompilation** no

**R topics documented:**

- dagLogo-package ........................................... 2  
- buildBackgroundModel .................................. 2  
- colorsets .................................................. 4  
- dagBackground-class ...................................... 4  
- dagHeatmap ............................................... 5  
- dagLogo .................................................... 5  
- dagPeptides-class ........................................ 6  
- ecoli.proteome ............................................ 7  
- fetchSequence ............................................. 8  
- formatSequence ........................................... 9  
- nameHash ................................................. 10  
- prepareProteome .......................................... 10  
- Proteome-class ........................................... 11  
- proteome.example ......................................... 12  
- seq.example ............................................... 12  
- testDAU .................................................... 13  
- testDAUresults-class .................................... 14  

Index 15
**Description**

We implement iceLogo by R to visualize significant conserved amino acid sequence pattern based on probability theory. Compare to iceLogo, dagLogo can also visualize significant sequence patterns by clustering the peptides by groups such as charge, chemistry, hydrophobicity and etc.

**Details**

<table>
<thead>
<tr>
<th>Package</th>
<th>dagLogo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Date</td>
<td>2013-09-31</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;= 2)</td>
</tr>
</tbody>
</table>

DAG: Differential Amino acid Group

There are several differences between dagLogo from iceLogo:

1. The sequence patterns can be grouped by charge, chemistry, hydrophobicity and etc.
2. dagLogo accepts different length of aligned amino acid sequences.
3. Except Random, regional (called restricted in dagLogo) and terminal (called anchored) background model, the background sequence could be set to other regions of the genes in inputs and complementary set of the proteome.

**Author(s)**

Jianhong Ou, Julie Lihua Zhu

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

**Examples**

```r
data("seq.example")
data("proteome.example")
bg <- buildBackgroundModel(seq.example, proteome=proteome.example, permutationSize=10L)
t <- testDAU(seq.example, bg)
dagLogo(t)
```

**Description**

build background model for dag test
Usage

buildBackgroundModel(dagPeptides,
  bg=c("wholeGenome", "inputSet", "nonInputSet"),
  model=c("any", "anchored"),
  targetPosition=c("any", "Nterminus", "Cterminus"),
  uniqueSeq=TRUE,
  permutationSize=30L,
  rand.seed=1,
  replacement=FALSE,
  proteome)

Arguments

dagPeptides   an object of dagPeptides, output of `fetchSequence` or `formatSequence`
bg            could be "wholeGenome", "inputSet" or "nonInputSet"
model         could be "any" or "anchored"
targetPosition could be "any", "Nterminus" or "Cterminus"
uniqueSeq     should the background sequence be unique?
permutationSize how many times should it samples
rand.seed      random seed
replacement    Should sampling be with replacement?
proteome       an object of Proteome, output of `prepareProteome`

Details

The background could be generated from wholeGenome, inputSet or nonInputSet. whole genome: randomly select subsequences from the whole genome with each subsequence containing amino acids with same width of input sequences. anchored whole genome: randomly select subsequences from the whole genome with each subsequence containing amino acids with same width of input sequences where the middle amino acids must contain anchor amino acid, e.g., K, which is specified by user. input set: same to whole genome, but only use protein sequence from input id and not including the site specified in input sequences anchored input set: same to anchored whole genome, but only use protein sequences from input id, and not including the site specified in input sequences. non-input set: whole genome - input set. anchored non-input set: whole genome - input set and the middle amino acids must contain anchor amino acid.

Value

an object of dagBackground which contains background and permutationSize.

Author(s)

Jianhong Ou, Alexey Stukalov, Julie Zhu

See Also

`prepareProteome`
Examples

data("seq.example")
data("proteome.example")
bg <- buildBackgroundModel(seq.example, proteome=proteome.example)

colorsets

retrieve color setting for logo

Description

retrieve prepared color setting for logo

Usage

colorsets(colorScheme=c("null", "classic", "charge", "chemistry", "hydrophobicity"))

Arguments

colorScheme could be 'null', 'charge', 'chemistry', 'classic' or 'hydrophobicity'

Value

A character vector of color scheme

Author(s)

Jianhong Ou

Examples

col <- colorsets("hydrophobicity")

dagBackground-class

Class "dagBackground"

Description

An object of class "dagBackground" represents background model.

Objects from the Class

Objects can be created by calls of the form new("dagBackground", background, permutationSize).

Slots

background Object of class "list" records the background model
permutationSize code "integer" permutation size of background
**dagHeatmap**

plot heatmap for test results

**Description**

plot heatmap for test results

**Usage**

dagHeatmap(testDAUresults, type=c("diff", "zscore"), ...)

**Arguments**

testDAUresults output of testDAU, should be an object of testDAUresults
type "diff" or "zscore"
... parameter could be passed to pheatmap

**Value**

none

**Author(s)**

Jianhong Ou

**Examples**

```r
data("seq.example")
data("proteome.example")
bg <- buildBackgroundModel(seq.example, proteome=proteome.example, permutationSize=10)
t <- testDAU(seq.example, bg)
dagHeatmap(t)
```

---

**dagLogo**

plot sequence logo for test results

**Description**

plot sequence logo for test results

**Usage**

dagLogo(testDAUresults, type=c("diff", "zscore"), pvalueCutoff=0.05, namehash=NULL, font="Helvetica-Bold", textgp=gpar(), legend=FALSE, labelRelativeToAnchor=FALSE, labels=NULL)
Arguments

- **testDAUresults**: output of `testDAU`, should be an object of `testDAUresults` type
- **type**: "diff" or "zscore"
- **pvalueCutoff**: pvalue cutoff for logo plot
- **namehash**: the hash table to convert rownames of test results to a single letter to be plotted in the logo
- **font**: font for logo symbol
- **textgp**: text parameter
- **legend**: plot legend or not, default false.
- **labelRelativeToAnchor**: plot label relative to anchor or not, default false.
- **labels**: the labels in each position.

Value

- none

Author(s)

- Jianhong Ou

See Also

- **nameHash**

Examples

```r
data("seq.example")
data("proteome.example")
bg <- buildBackgroundModel(seq.example, proteome=proteome.example, permutationSize=10)
t <- testDAU(seq.example, bg)
dagLogo(t)
```

```
dagPeptides-class  Class "dagPeptides"
```

Description

An object of class "dagPeptides" represents the information of peptides.

Objects from the Class

Objects can be created by calls of the form `new("dagPeptides", data, peptides, upstreamOffset, downstreamOffset,...)"`.
ecoli.proteome

Slots

data Object of class "data.frame" The details of the input sequences. It includes the columns: IDs, anchorAA (anchor Amino Acid), anchorPos (anchor Position), peptide (protein peptide), anchor, upstream, downstream (peptides in given upstream and downstream offset from anchor)

peptides code"matrix" The input peptides. Each column contains one peptide in that position

upstreamOffset "numeric" The upstream offset from anchor
downstreamOffset "numeric" The downstream offset from anchor
type "character" ID type of inputs

ecoli.proteome the subset proteome of Escherichia coli

Description
the subset proteome of Escherichia coli

Usage
data(ecoli.proteome)

Format
An object of Proteome for Escherichia coli proteome. The format is: A list with one data frame and an character.

proteome 'data.frame': obs. of 4 variables
type 'character': "UniProt"

The format of proteome is

ENTREZ_GENE a character vector, records entrez gene id
SEQUENCE a character vector, peptide sequences
ID a character vector, Uniprot ID
LEN a character vector, length of peptides

Details
used in the examples Annotation data obtained by: library(UniProt.ws) taxId(UniProt.ws) <- 562 proteome <- prepareProteome(UniProt.ws, species="Escherichia coli")

Examples
data(ecoli.proteome)
head(ecoli.proteome@proteome)
ecoli.proteome@type
fetchSequence

fetch sequence by id

Description
fetch amino acid sequence by given identifiers via biomaRt or proteome prepared by prepareProteome

Usage
fetchSequence(IDs, type="entrezgene", anchorAA=NULL, anchorPos, mart, proteome, upstreamOffset, downstreamOffset)

Arguments
IDs A vector of Identifiers to retrieve peptides
type type of identifiers
anchorAA a vector of character, anchor Amino Acid
anchorPos a vector of character or numeric, anchor position, for example, K121. Or a vector of character with amino acid sequences. If AA sequences is used, the anchorAA must be the a vector of character with single AA for each.
mart an object of Mart
proteome an object of Proteome, output of prepareProteome
upstreamOffset an integer, upstream offset position
downstreamOffset an integer, downstream offset position

Value
return an object of dagPeptides

Author(s)
Jianhong Ou, Alexey Stukalov, Julie Zhu

See Also
formatSequence

Examples
if(interactive()){  
mart <- useMart("ensembl", "dmelanogaster_gene_ensembl")  
dat <- read.csv(system.file("extdata", "dagLogoTestData.csv", package="dagLogo"))  
seq <- fetchSequence(as.character(dat$entrez_geneid[1:5]),
  anchorPos=as.character(dat$NCBI_site[1:5]),
  mart=mart,
  upstreamOffset=7,
  downstreamOffset=7)
  ## sample: use sequence as anchorPos
sequences <- seq@peptides
sequences[, 8] <- "k"
```r
sequences <- apply(sequences, 1, paste, collapse"
seq <- fetchSequence(as.character(seq@data$IDs),
  anchorAA="k",
  anchorPos=sequences,
  mart=mart,
  upstreamOffset=7,
  downstreamOffset=7)
## sample: use sequence as anchorPos 2
sequences <- cbind(seq@peptides[, 1:8], ",", seq@peptides[, 9:15])
seq <- fetchSequence(as.character(seq@data$IDs),
  anchorAA="*",
  anchorPos=sequences,
  mart=mart,
  upstreamOffset=7,
  downstreamOffset=7)
```
prepareProteome

Examples

```r
if(interactive()){
  dat <- unlist(read.delim(system.file("extdata", "grB.txt", package="dagLogo"), header=F, as.is=TRUE))
  proteome <- prepareProteome(fasta=system.file("extdata", "HUMAN.fasta", package="dagLogo"))
  seq <- formatSequence(dat, proteome)
}
```

descHash

convert group name to a single character

Description

convert group name to a single character to shown in a logo

Usage

```r
nameHash(nameScheme=c("classic", "charge", "chemistry", "hydrophobicity"))
```

Arguments

- `nameScheme`
  - could be "classic", "charge", "chemistry", "hydrophobicity"

Value

A character vector of name scheme

Author(s)

Jianhong Ou

Examples

```r
nameHash("charge")
```

prepareProteome

prepare proteome for background building

Description

prepare proteome from UniProt webserver or a fasta file

Usage

```r
prepareProteome(UniProt.ws, fasta, species="unknown")
```
Proteome-class

Arguments

- **UniProt.ws**: an object of UniProt.ws
- **fasta**: fasta file name or an object of AAStringSet
- **species**: an character to assign the species of the proteome

Value

- an object of Proteome which contain protein sequence information

Author(s)

Jianhong Ou

See Also

- formatSequence, buildBackgroundModel

Examples

```r
if(interactive()){
  library(UniProt.ws)
  UniProt.ws <- UniProt.ws(taxId=7227)
  proteome <- prepareProteome(UniProt.ws, species="Drosophila melanogaster")
}
```

---

Proteome-class  
Class "Proteome"

Description

An object of class "Proteome" represents proteome of a given species.

Objects from the Class

Objects can be created by calls of the form `new("Proteome", proteome, type, species)`.

Slots

- **proteome**: Object of class "data.frame" the proteome of a given species, should include ids and peptide sequences.
- **type**: code"character" indicates how the object is prepared, could be "fasta" or "UniProt"
- **species**: "character" the species
proteome.example  
the subset proteome of fruit fly

Description
the subset proteome of fruit fly

Usage
data(proteome.example)

Format
An object of Proteome for fly subset proteome. The format is: A list with one data frame and an character.

proteome 'data.frame': 1406 obs. of 4 variables
type 'character': "UniProt"

The format of proteome is

ENTREZ_GENE a character vector, records entrez gene id
SEQUENCE a character vector, peptide sequences
ID a character vector, Uniprot ID
LEN a character vector, length of peptides

Details
used in the examples Annotation data obtained by: library(UniProt.ws) taxId(UniProt.ws) <- 7227
proteome <- prepareProteome(UniProt.ws) proteome@proteome <- proteome@proteome[sample(1:19902, 1406), ]

Examples

data(proteome.example)
head(proteome.example@proteome)
proteome.example@type

seq.example  
example object of dagPeptides

Description
example object of dagPeptides

Usage
data(seq.example)
testDAU

Format

An object of dagPeptides. The format is: A list.

data 'data.frame': 732 obs. of 7 variables
peptides 'matrix': amnio acid in each position
upstreamOffset an integer, upstream offset position
downstreamOffset an integer, downstream offset position
type "character", type of identifiers

The format of data is

IDs a character vector, input identifiers
anchorAA a character vector, anchor amino acid provided in inputs
anchorPos a numeric vector, anchor position in the protein
peptide a character vector, peptide sequences
anchor a character vector, anchor amino acid in the protein
upstream a character vector, upstream peptides
downstream a character vector, downstream peptides

Details

used in the examples seq obtained by: mart <- useMart("ensembl","dmelanogaster_gene_ensembl")
dat <- read.csv(system.file("extdata", "dagLogoTestData.csv", package="dagLogo")) seq <- fetchSequence(as.character(dat$entrez_geneid), anchorPos=as.character(dat$NCBI_site), mar=t, upstreamOffset=7, downstreamOffset=7)

Examples

data(seq.example)
head(seq.example@peptides)
seq.example@upstreamOffset
seq.example@downstreamOffset

desc FAT UST

Description

Performs DAU test

Usage

testDAU(dagPeptides, dagBackground, 
        group=c("null", "classic", "charge", "chemistry", "hydrophobicity"),
        bgNoise=NA)
testDAUresults-class

**Arguments**

- *dagPeptides*: an object of `dagPeptides`, output of `fetchSequence` or `formatSequence`
- *dagBackground*: an object of `dagBackground`, output of `buildBackgroundModel`
- *group*: could be "null", "classic", "charge", "chemistry", "hydrophobicity"
- *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**

an object of testDAUresults ready for plotting

**Author(s)**

Jianhong Ou, Alexey Stukalov, Julie Zhu

**Examples**

```r
data("seq.example")
data("proteome.example")
bgs <- buildBackgroundModel(seq.example, proteome=proteome.example)
t <- testDAU(seq.example, bgs, bgNoise=0.05)
```

---

testDAUresults-class  Class "testDAUresults"

**Description**

An object of class "testDAUresults" represents background model.

**Objects from the Class**

Objects can be created by calls of the form `new("dagBackground", group="character", difference="matrix", zscore="matrix", pvalue="matrix", background="matrix", motif="matrix", upstream="numeric", downstream="numeric")`.

**Slots**

- *group*: Object of class "character" could be "null", "classic", "charge", "chemistry", "hydrophobicity"
- *difference*: code"matrix" the difference of inputs from background for each amino acid in each position
- *zscore*: code"matrix" z score for each amino acid in each position
- *pvalue*: code"matrix" pvalue for each amino acid in each position
- *background*: code"matrix" background frequencies for each amino acid in each position
- *motif*: code"matrix" inputs frequencies for each amino acid in each position
- *upstream*: "numeric" The upstream offset from anchor
- *downstream*: "numeric" The downstream offset from anchor
Index

*Topic **classes**
  
  dagBackground-class, 4
  dagPeptides-class, 6
  Proteome-class, 11
  testDAUresults-class, 14

*Topic **datasets**
  
  ecoli.proteome, 7
  proteome.example, 12
  seq.example, 12

*Topic **figure**
  
  colorsets, 4
  dagHeatmap, 5
  dagLogo, 5
  nameHash, 10

*Topic **misc**
  
  buildBackgroundModel, 2
  fetchSequence, 8
  formatSequence, 9
  prepareProteome, 10
  testDAU, 13

*Topic **package**
  
  dagLogo-package, 2

buildBackgroundModel, 2, 11, 14

colorsets, 4

dagBackground-class, 4

dagHeatmap, 5

dagLogo, 5

dagLogo-package, 2

dagPeptides, 8

dagPeptides-class, 6

ecoli.proteome, 7

fetchSequence, 3, 8, 9, 14

formatSequence, 3, 8, 9, 11, 14

nameHash, 6, 10

prepareProteome, 3, 8, 9, 10

Proteome-class, 11

proteome.example, 12

seq.example, 12

testDAU, 5, 6, 13

testDAUresults-class, 14