Package ‘ecolitk’

January 14, 2017

Version 1.46.0

Date 2010-04

Title Meta-data and tools for E. coli

Author Laurent Gautier

Maintainer Laurent Gautier <lgautier@gmail.com>

biocViews Annotation, Visualization

Depends R (>= 2.10)

Imports Biobase, graphics, methods

Suggests ecoliLeucine, ecolicdf, graph, multtest, affy

Description Meta-data and tools to work with E. coli. The tools are mostly plotting functions to work with circular genomes. They can used with other genomes/plasmids.

License GPL (>= 2)

NeedsCompilation no

R topics documented:

ecoli.m52.genome .................................................. 2
ecoligenome.operon ................................................. 3
ecoligenomeBNUM ................................................... 4
ecoligenomeBNUM2MULTIFUN ..................................... 4
gcontent ............................................................... 5
linkedmultiget ....................................................... 5
multiFun ............................................................... 6
pointsCircle .......................................................... 7
polar2xy ............................................................... 8
polygonChrom ......................................................... 9
wstringapply .......................................................... 11

Index 13
**Description**

Meta-data related to *Escherichia coli*

**Usage**

```r
data(ecoli.m52.genome)
data(ecoligenomeCHRLOC)
data(ecoligenomeSYMBOL2AFFY)
data(ecoligenomeSYMBOL)
data(ecoligenomeSTRAND)
data(ecoligenome.operon)
ecoli.len
```

**Format**

The format for `ecoli.m52.genome` is character with genome sequence. The format for `ecoligenomeCHRLOC` is an environment (as a hash table). Each key is an Affymetrix probe set ID, and each value is a vector of two integers (beginning and end - see the details below). The format for `ecoligenomeSYMBOL2AFFY` is an environment (as a hash table). Each key is a gene symbol name. The format for `ecoligenomeSYMBOL` is an environment (as a hash table). Each key is an Affymetrix probe set ID. `ecoli.len` is a variable containing the size of the genome in `ecoli.m52.genome`.

**Details**

The environments `ecoligenomeSYMBOL2AFFY` and `ecoligenomeSYMBOL` are like the ones in the data packages built by `annBuilder`.

The environment `ecoligenomeCHRLOC` differs: two integers are associated with each key, one corresponds to the beginning of the segment the other to the end.

The environment `ecoligenomeSTRAND` returns a logical. TRUE means that the orientation is ‘+’, FALSE means that the orientation is ‘-’ (and NA is used when irrelevant for the key).

**Source**


**Examples**

```r
data(ecoli.m52.genome)
```
The known operon in the Escherichia coli genome.

Usage

data(ecoligenome.operon)

Format

A data frame with 932 observations (genes) on the following 4 variables.

- **gene.name**  a character vector
- **gene.annotation**  a character vector
- **operon.name**  a factor with levels the names of the operons
- **operon.comments**  a factor with levels the comments for the operons

Details

For some operons, the source of information specifies the existence of regulating elements such as promoter, terminator, box, etc... In those cases, the gene.name is set to "Regulation", and the gene.annotation gives what kind of regulating element it is. If volunteers, it would be neat to map those on the genome... Besides that, not much to add. The data structure is fairly straightforward.

Source

Built from the webpage: [http://www.cib.nig.ac.jp/dda/backup/taitoh/ecoli.operon.html](http://www.cib.nig.ac.jp/dda/backup/taitoh/ecoli.operon.html)

Examples

```r
library(Biobase)
data(ecoligenome.operon)
data(ecoligenomeSYMBOL2AFFY)

## something that might be useful when working with Affymetrix data:
## get the Affymetrix identifiers for the probe sets bundled in operons
## (see the vignette for more details)
ecoligenome.operon$saffyid <-
unname(unlist(mget(ecoligenome.operon$gene.name,
                   ecoligenomeSYMBOL2AFFY, ifnotfound=NA)))
```
ecoligenomeBNUM2MULTIFUN

ecoligenomeBNUM  Environment for 'bnum' identifiers

Description

Environments to associate Affymetrix probe set IDs with 'bnum' IDs

Usage

data(ecoligenomeBNUM)
data(ecoligenomeBNUM2SYMBOL)
data(ecoligenomeBNUM2ENZYME)
data(ecoligenomeBNUM2GENBANK)
data(ecoligenomeBNUM2GENEPRODUCT)
data(ecoligenomeSYMBOL2BNUM)

Format

These are environment objects.

Details

Escherichia coli genes are sometimes identified by 'bnum's. This identifier is typically a 'b' followed by digits.

Source

BNUM numbers were parsed out of the Affymetrix identifiers. BNUM2* were obtained from the GenProtEC website.

ecoligenomeBNUM2MULTIFUN  Environment

Description

An environment to store associations between 'bnum' identifiers (key) and 'MultiFun' identifiers (or strand information).

Usage

data(ecoligenomeBNUM2MULTIFUN)

Format

The format is: length 0 <environment> - attr(*, "comments")= chr "GenProtEC: MultiFun assignments for E. coli modules September 17th, 2003"
Details

'MultiFun' is a classification scheme. The structure is 'approximately tree-like'. Several 'MultiFun' numbers can be assigned to one 'bnum'.

Source

"http://genprotec.mbl.edu/files/MultiFun.txt"

gccontent  function to compute gccontent

Description

A simple R function to compute the GC content of a sequence

Usage

gccontent(x)

Arguments

x a vector of mode character

Details

This a simple (and not particularly fast) function to compute the GC content of sequence. When speed is an issue, one should use the function in the package matchprobes. This function only exists to avoid dependency on this package.

Value

The GC content (numeric)

linkedmultiget  A function to look for values across linked environments

Description

A function to look for values across linked environments.

Usage

linkedmultiget(x, envir.list = list(), unique = TRUE)

Arguments

x The keys in the first environment in the list.
envir.list A list of environments.
unique Simplify the list returned by ensuring that the values for each key are unique.
Details

Environments can be considered as hashtables. The keys are obviously strings, but in some cases
the associated values are also strings. This is the case for annotation environments (as built with
the package AnnBuilder). This function helps to look for values across several environments: the
keys have associated values in a first environment, these values are used as keys in the second
environments, etc...

Value

A list of length the length of x.

Author(s)

Laurent Gautier

See Also

mget

Examples

data(ecoligenomeBNUM)
data(ecoligenomeBNUM2MULTIFUN)
data(multiFun)

## get 5 Affymetrix IDs
set.seed(456)
my.affyids <- sample(ls(ecoligenomeBNUM), 5)

## get the MULTIFUN annotations for them
r <- linkedmultiget(my.affyids, list(ecoligenomeBNUM,
                                    ecoligenomeBNUM2MULTIFUN, multiFun))

print(r)

---

multiFun  


title

Description

The MultiFun classification scheme

Usage

data(multiFun)
data(ecoligenomeMULTIFUN2GO)

Format

These are environments.
pointsCircle

Source
http://genprotec.mbl.edu/files/MultiFun.txt

Examples
## To be done...

pointsCircle  

Functions to plot circular related figures

Description
Functions to plot circular related figures

Usage

```r
linesCircle(radius, center.x = 0, center.y = 0, edges = 300, ...)
polygonDisk(radius, center.x = 0, center.y = 0, edges=300, ...)
arrowsArc(theta0, theta1, radius, center.x = 0, center.y = 0, edges = 10, length = 0.25, angle = 30, code = 2, ...)
pointsArc(theta0, theta1, radius, center.x = 0, center.y = 0, ...)
linesArc(theta0, theta1, radius, center.x = 0, center.y = 0, ...)
polygonArc(theta0, theta1, radius.in, radius.out, center.x = 0, center.y = 0, edges = 10, col = "black", border = NA, ...)
```

Arguments

- `theta0, theta1`: start and end angles for the arc
- `radius`: radius of the circle
- `radius.in`: inner radius
- `radius.out`: outer radius
- `center.x, center.y`: Coordinates for the center of the circle (default to (0, 0))
- `edges`: number of edges the shape is made of
- `col`: color
- `border`: border (see `polygon`)
- `length, angle, code`: see the corresponding parameters for the function `arrows`
- `...`: optional graphical parameters

Details
Details to come...for now the best to run the examples and experiment by yourself...
Value

Function only used for their border effects.

Author(s)

laurent

Examples

```r
par(mfrow=c(2,2))
n <- 10
thetas <- rev(seq(0, 2 * pi, length=n))
rhos <- rev(seq(1, n) / n)

xy <- polar2xy(rhos, thetas)
colo <- heat.colors(n)

plot(0, 0, xlim=c(-2, 2), ylim=c(-2, 2), type="n")
for (i in 1:n)
  linesCircle(rhos[i]/2, xy$x[i], xy$y[i])

plot(0, 0, xlim=c(-2, 2), ylim=c(-2, 2), type="n")
for (i in 1:n)
  polygonDisk(rhos[i]/2, xy$x[i], xy$y[i], col=colo[i])

plot(0, 0, xlim=c(-2, 2), ylim=c(-2, 2), type="n", xlab="", ylab="")
for (i in 1:n)
  polygonArc(0, thetas[i],
              rhos[i]/2, rhos[i],
              center.x = xy$x[i], center.y = xy$y[i], col=colo[i])

plot(0, 0, xlim=c(-2, 2), ylim=c(-2, 2), type="n", xlab="", ylab="")
for (i in (1:n)[-1]) {
  linesCircle(rhos[i-1], col="gray", lty=2)
  polygonArc(thetas[i-1], thetas[i],
             rhos[i-1], rhos[i], col=colo[i],
             edges=20)
  arrowsArc(thetas[i-1], thetas[i],
            rhos[i] + 1, col=colo[i],
            edges=20)
}
```

---

polar2xy

*Functions to perform polar coordinate related functions*

Description

Functions to perform polar coordinate related functions
Usage

polar2xy(rho, theta)
xy2polar(x, y)
rotate(x, y, alpha)

Arguments

x  cartesian coordinate
y  cartesian coordinate
rho polar radius rho
theta polar angle theta
alpha angle to perform rotation

Details

y and theta can be respectively missing. In this case, x and rho are expected to be lists with entries x, y, rho, theta respectively.

Examples

n <- 40
nn <- 2
thetas <- seq(0, nn * 2 * pi, length=n)
rhos <- seq(1, n) / n
plot(c(-1, 1), c(-1, 1), type="n")
abline(h=0, col="grey")
abline(v=0, col="grey")
xy <- polar2xy(rhos, thetas)
points(xy$x, xy$y, col=rainbow(n))

polygonChrom  

Functions to plot circular chromosomes informations

Description

Functions to plot circular chromosomes informations

Usage

cPlotCircle(radius=1, xlim=c(-2, 2), ylim=xlim, edges=300, main=NULL, main.inside, ...)

chromPos2angle(pos, len.chrom, rot=pi/2, clockwise=TRUE)
polygonChrom\( (\text{begin}, \text{end}, \text{len.chrom}, \text{radius.in}, \text{radius.out}, \) \\
\text{total.edges} = 300, \) \\
\text{edges = max(round(abs(\text{end} - \text{begin})/\text{len.chrom} \times \text{total.edges}), 2, \text{na.rm} = \text{TRUE}), \) \\
\text{rot = pi/2, clockwise = \text{TRUE}, ...} \)

linesChrom(\text{begin}, \text{end}, \text{len.chrom}, \text{radius}, \\
\text{total.edges} = 300, \) \\
\text{edges = max(round(abs(\text{end} - \text{begin})/\text{len.chrom} \times \text{total.edges}), 2, \text{na.rm} = \text{TRUE}), \) \\
\text{rot = pi/2, clockwise = \text{TRUE}, ...} \)

ecoli.len

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>radius</td>
<td>radius</td>
</tr>
<tr>
<td>xlim, ylim</td>
<td>range for the plot. Can be used to zoom-in a particular region.</td>
</tr>
<tr>
<td>pos</td>
<td>position (nucleic base coordinate)</td>
</tr>
<tr>
<td>begin</td>
<td>beginning of the segment (nucleic base number).</td>
</tr>
<tr>
<td>end</td>
<td>end of the segment (nucleic base number).</td>
</tr>
<tr>
<td>len.chrom</td>
<td>length of the chromosome in base pairs</td>
</tr>
<tr>
<td>radius.in</td>
<td>inner radius</td>
</tr>
<tr>
<td>radius.out</td>
<td>outer radius</td>
</tr>
<tr>
<td>total.edges</td>
<td>total number of edges for the chromosome</td>
</tr>
<tr>
<td>edges</td>
<td>number of edges for the specific segment(s)</td>
</tr>
<tr>
<td>rot</td>
<td>rotation (default is pi / 2, bringing the angle zero at 12 o’clock)</td>
</tr>
<tr>
<td>clockwise</td>
<td>rotate clockwise. Default to TRUE.</td>
</tr>
<tr>
<td>main, main.inside</td>
<td>main titles for the plot</td>
</tr>
<tr>
<td>...</td>
<td>optional graphical parameters</td>
</tr>
</tbody>
</table>

**Details**

The function chromPos2angle is a convenience function. The variable ecoli.len contains the size of the Escheria coli genome considered (K12).

**Value**

Except chromPos2angle, the function are solely used for their border effects.

**Author(s)**

laurent <laurent@cbs.dtu.dk>
### Examples

```r
data(ecoligenomeSYMBOL2AFFY)
data(ecoligenomeCHRLOC)

## find the operon lactose ("lac" genes)
lac.i <- grep("^lac", ls(ecoligenomeSYMBOL2AFFY))
lac.symbol <- ls(ecoligenomeSYMBOL2AFFY)[lac.i]
lac.affy <- unlist(lapply(lac.symbol, get, envir=ecoligenomeSYMBOL2AFFY))

beg.end <- lapply(lac.affy, get, envir=ecoligenomeCHRLOC)
beg.end <- matrix(unlist(beg.end), nc=2, byrow=TRUE)

lac.o <- order(beg.end[, 1])
lac.i <- lac.i[lac.o]
lac.symbol <- lac.symbol[lac.o]
lac.affy <- lac.affy[lac.o]
beg.end <- beg.end[lac.o, ]

lac.col <- rainbow(length(lac.affy))

par(mfrow=c(2,2))

## plot
cPlotCircle(main="lac genes")
polygonChrom(beg.end[, 1], beg.end[, 2], ecoli.len, 1, 1.2, col=lac.col)
rect(0, 0, 1.1, 1.1, border="red")

cPlotCircle(xlim=c(0, 1.2), ylim=c(0, 1.1))
polygonChrom(beg.end[, 1], beg.end[, 2], ecoli.len, 1, 1.1, col=lac.col)
rect(0.4, 0.8, 0.7, 1.1, border="red")

cPlotCircle(xlim=c(.45, .5), ylim=c(.85, 1.0))
polygonChrom(beg.end[, 1], beg.end[, 2], ecoli.len, 1, 1.03, col=lac.col)

mid.genes <- apply(beg.end, 1, mean)
mid.angles <- chromPos2angle(mid.genes, ecoli.len)
xy <- polar2xy(1.03, mid.angles)
xy.labels <- data.frame(x = seq(0.45, 0.5, length=4), y = seq(0.95, 1.0, length=4))
segments(xy$x, xy$y, xy.labels$x, xy.labels$y, col=lac.col)
text(xy.labels$x, xy.labels$y, lac.symbol, col=lac.col)
```

---

**wstringapply**

*Apply a function on a window sliding on a string*

### Description

Apply a function on a window sliding on a string.

### Usage

```r
wstringapply(x, SIZE, SLIDE, FUN, ...)
```
**Arguments**

- **x**  The string
- **SIZE**  The size of the window (number of characters).
- **SLIDE**  offset to move at each slide
- **FUN**  The function to be applied
- ...  optional parameter for the function FUN

**Details**

Apply the function FUN to substrings of x of length SIZE.

**Value**

A list of size \( nchar(x) - SIZE \).

**Author(s)**

L, Gautier
Index

*Topic** aplot
  pointsCircle, 7

*Topic** datasets
  ecoli.m52.genome, 2
  ecoligenome.operon, 3
  ecoligenomeBNUM, 4
  ecoligenomeBNUM2MULTIFUN, 4
  multifun, 6

*Topic** dplot
  polygonChrom, 9

*Topic** hplot
  polygonChrom, 9

*Topic** manip
  gccontent, 5
  linkedmultitiget, 5
  polar2xy, 8
  wstringapply, 11
  arrows, 7
  arrowsArc(pointsCircle), 7
  chromPos2angle(polygonChrom), 9
  cPlotCircle(polygonChrom), 9
  ecoli.len(ecoli.m52.genome), 2
  ecoli.m52.genome, 2
  ecoli.operon(ecoli.m52.genome), 2
  ecoligenome.operon, 3
  ecoligenomeBNUM, 4
  ecoligenomeBNUM2ENZYME
    (ecoligenomeBNUM), 4
  ecoligenomeBNUM2GENBANK
    (ecoligenomeBNUM), 4
  ecoligenomeBNUM2GENPRODUCT
    (ecoligenomeBNUM), 4
  ecoligenomeBNUM2MULTIFUN, 4
  ecoligenomeBNUM2STRAND
    (ecoligenomeBNUM2MULTIFUN), 4
  ecoligenomeBNUM2SYMBOL
    (ecoligenomeBNUM), 4
  ecoligenomeCHRLOC(ecoli.m52.genome), 2
  ecoligenomeMULTIFUN2GO(multiFun), 6
  ecoligenomeSTRAND(ecoli.m52.genome), 2
  ecoligenomeSYMBOL(ecoli.m52.genome), 2
  ecoligenomeSYMBOL2AFFY
    (ecoli.m52.genome), 2
  ecoligenomeSYMBOL2BNUM
    (ecoligenomeBNUM), 4
  gccontent, 5
  linesArc(pointsCircle), 7
  linesChrom(polygonChrom), 9
  linesCircle(pointsCircle), 7
  linkedmultitiget, 5
  mget, 6
  multiFun, 6
  pointsArc(pointsCircle), 7
  pointsCircle, 7
  polar2xy, 8
  polygon, 7
  polygonArc(pointsCircle), 7
  polygonChrom, 9
  polygonDisk(pointsCircle), 7
  rotate(polar2xy), 8
  wstringapply, 11
  xy2polar(polar2xy), 8