

Package ‘GOplot’

January 30, 2015

Type Package

Title Visualization Of Functional Analysis Data

Version 1.0

Date 2015-01-27

Author Wencke Walter, Fatima Sanchez-Cabo

Maintainer Wencke Walter <wencke.walter@arcor.de>

Description Implementation of multilayered visualizations for enhanced graphical representation of functional analysis data. It combines and integrates omics data derived from expression and functional annotation enrichment analyses. Its plotting functions have been developed with an hierarchical structure in mind: starting from a general overview to identify the most enriched categories (modified bar plot, bubble plot) to a more detailed one displaying different types of relevant information for the molecules in a given set of categories (circle plot, chord plot, cluster plot).

Depends ggplot2, ggdendro, grid, gridExtra, RColorBrewer, R (>= 3.0.3)

License GPL-2

Suggests knitr

VignetteBuilder knitr

LazyData TRUE

NeedsCompilation no

Repository CRAN

Date/Publication 2015-01-30 12:56:15

R topics documented:

bezier	2
chord_dat	2
circleFun	3
circle_dat	4
draw_table	5
EC	5

get_overlap	6
get_overlap2	6
GOBar	7
GOBubble	8
GOChord	9
GOCircle	10
GOCluster	12
GOVenn	13
theme_blank	14

Index 16

bezier	<i>bezier</i>
--------	---------------

Description

bezier

Usage

```
bezier(data, process.col)
```

Arguments

data	data frame x.start,x.end,y.start,y.end
process.col	Color of the processes

chord_dat	<i>Creates a binary matrix.</i>
-----------	---------------------------------

Description

The function creates a matrix which represents the binary relation (1= is related to, 0= is not related to) between selected genes (row) and processes (column). The resulting matrix can be visualized with the [GOChord](#) function.

Usage

```
chord_dat(data, genes, process)
```

Arguments

data	A data frame with columns for GO ID/term and genes.
genes	A character vector of selected genes OR data frame with columns for gene ID and logFC.
process	A character vector of selected processes

Details

If more than one logFC value for each gene is at disposal, only one should be used to create the binary matrix. The other values have to be added manually later.

Value

A binary matrix

See Also

[GOChord](#)

Examples

```
## Not run:  
#Load the included dataset  
data(EC)  
  
#Building the circ object  
circ<-circular_dat(EC$david, EC$genelist)  
  
#Building the binary matrix  
chord<-chord_dat(circ, EC$genes, EC$process)  
  
## End(Not run)
```

circleFun

circleFun

Description

circleFun

Usage

```
circleFun(center = c(0, 0), diameter = 1, npoints = 100)
```

Arguments

center	Center point of the circle
diameter	Defines the diameter of the circle
npoints	Number of points used to draw the circle

circle_dat	<i>Creates a plotting object.</i>
------------	-----------------------------------

Description

The function takes the results from a functional analysis (for example DAVID) and combines it with a list of selected genes and their logFC. The resulting data frame can be used as an input for various plotting functions.

Usage

```
circle_dat(terms, genes)
```

Arguments

terms	A data frame with columns for 'category', 'ID', 'term', adjusted p-value ('adj_pval') and 'genes'
genes	A data frame with columns for 'ID', 'logFC'

Details

Since most of the gene- annotation enrichment analysis are based on the gene ontology database the package was build with this structure in mind, but is not restricted to it. Gene ontology is structured as an acyclic graph and it provides terms covering different areas. These terms are grouped into three independent categories: BP (biological process), CC (cellular component) or MF (molecular function).

The "ID" and "term" columns of the terms data frame refer to the ID and term description, whereas the ID is optional.

The "ID" column of the genes data frame can contain any unique identifier. Nevertheless, the identifier has to be the same as in "genes" from terms.

Examples

```
## Not run:  
#Load the included dataset  
data(EC)  
  
#Building the circ object  
circ<-circular_dat(EC$david, EC$genelist)  
  
## End(Not run)
```

draw_table	<i>draw table</i>
------------	-------------------

Description

draw table

Usage

```
draw_table(data, size = 4, col = "")
```

Arguments

data	data frame; GO ID and GO term
size	numeric; defines font size
col	character; defines color of table header

EC	<i>Transcriptomic information of endothelial cells.</i>
----	---

Description

The data set contains the transcriptomic information of endothelial cells from two steady state tissues (brain and heart). More detailed information can be found in the paper by Nolan et al. 2013. The data was normalized and a statistical analysis was performed to determine differentially expressed genes. DAVID functional annotation tool was used to perform a gene- annotation enrichment analysis of the set of differentially expressed genes (adjusted p-value < 0.05).

Usage

```
data(EC)
```

Format

A list containing 5 items

Source

<http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE47067>

`get_overlap`*get_overlap*

Description`get_overlap`**Usage**`get_overlap(A, B, C)`**Arguments**

A Input list 1

B Input list 2

C Input list 3

`get_overlap2`*get_overlap2*

Description`get_overlap2`**Usage**`get_overlap2(A, B)`**Arguments**

A Input list 1

B Input list 2

GOBar	<i>Z-score colored barplot.</i>
-------	---------------------------------

Description

Z-score colored barplot of terms ordered alternatively by z-score or the negative logarithm of the adjusted p-value

Usage

```
GOBar(data, display, order.by.zscore = T, title, zsc.col)
```

Arguments

<code>data</code>	A data frame containing at least the term ID and/or term, the adjusted p-value and the z-score. A possible input can be generated with the <code>circle_dat</code> function
<code>display</code>	A character vector indicating whether a single plot ('single') or a facet plot with panels for each category should be drawn (default='single')
<code>order.by.zscore</code>	Defines the order of the bars. If TRUE the bars are ordered according to the z-scores of the processes. Otherwise the bars are ordered by the negative logarithm of the adjusted p-value
<code>title</code>	The title of the plot
<code>zsc.col</code>	Character vector to define the color scale for the z-score of the form <code>c(high, midpoint, low)</code>

Details

If `display` is used to facet the plot the width of the panels will be proportional to the length of the x scale.

Examples

```
## Not run:  
#Load the included dataset  
data(EC)  
  
#Building the circ object  
circ<-circular_dat(EC$david, EC$genelist)  
  
#Creating the bar plot  
GOBar(circ)  
  
#Faceting the plot  
GOBar(circ, display='multiple')  
  
## End(Not run)
```

GOBubble

Bubble plot.

Description

The function creates a bubble plot of the input data. The input data can be created with the help of the [circle_dat](#) function.

Usage

```
GOBubble(data, display, title, color, labels, ID = T, table.legend = T,
         table.col = T, bg = T)
```

Arguments

data	A data frame with columns for category, GO ID, term, adjusted p-value, z-score, count(num of genes)
display	A character vector. Indicates whether it should be a single plot ('single') or a facet plot with panels for each category (default='single')
title	The title (on top) of the plot
color	A character vector which defines the color of the bubbles for each category
labels	Sets a threshold for the displayed labels. The threshold refers to the -log(adjusted p-value) (default=5)
ID	If TRUE then labels are IDs else terms
table.legend	Defines whether a table of GO ID and GO term should be displayed on the right side of the plot or not (default=TRUE)
table.col	If TRUE then the table entries are colored according to their category, if FALSE then entries are black
bg	If TRUE within the plot a grey background will be drawn

Details

The x- axis of the plot represents the z-score. The negative logarithm of the adjusted p-value (corresponding to the significance of the term) is displayed on the y-axis. The area of the plotted circles is proportional to the number of genes assigned to the term. Each circle is colored according to its category and labeled alternatively with the ID or term name.

Examples

```
## Not run:
#Load the included dataset
data(EC)

#Building the circ object
circ<-circular_dat(EC$david, EC$genelist)
```



```

#Creating the bubble plot coloring the table entries according to the category
GOBubble(circ, table.col=T)

#Creating the bubble plot displaying the term instead of the ID and without the table
GOBubble(circ,ID=F,table.legend=F)

#Faceting the plot
GOBubble(circ, display='multiple')

## End(Not run)

```

GOChord

Displays the relationship between genes and terms.

Description

The GOChord function generates a circularly composited overview of selected/specific genes and their assigned processes or terms. More generally, it joins genes and processes via ribbons in an intersection-like graph. The input can be generated with the [chord_dat](#) function.

Usage

```

GOChord(data, title, space, gene.order, gene.size, gene.space, nlfc = 1,
        lfc.col, lfc.min, lfc.max, ribbon.col, border.size, process.label)

```

Arguments

<code>data</code>	The matrix represents the binary relation (1= is related to, 0= is not related to) between a set of genes (rows) and processes (columns); a column for the logFC of the genes is optional
<code>title</code>	The title (on top) of the plot
<code>space</code>	The space between the chord segments of the plot
<code>gene.order</code>	A character vector defining the order of the displayed gene labels
<code>gene.size</code>	The size of the gene labels
<code>gene.space</code>	The space between the gene labels and the segment of the logFC
<code>nlfc</code>	Defines the number of logFC columns (default=1)
<code>lfc.col</code>	The fill color for the logFC specified in the following form: c(color for low values, color for the mid point, color for the high values)
<code>lfc.min</code>	Specifies the minimum value of the logFC scale (default = -3)
<code>lfc.max</code>	Specifies the maximum value of the logFC scale (default = 3)
<code>ribbon.col</code>	The background color of the ribbons
<code>border.size</code>	Defines the size of the ribbon borders
<code>process.label</code>	The size of the legend entries

Details

The `gene.order` argument has three possible options: "logFC", "alphabetical", "none". Actually, the options are quite self-explanatory.

Another argument which needs a bit more explanation is `n1fc`. Differential expression analysis can be performed for multiple conditions and/or batches. Therefore, the data frame contains more than one logFC value per gene. To adjust to this situation the `n1fc` argument is used. It is a numeric value and it defines the number of logFC columns within the binary membership matrix. The default is "1" assuming that most of the time only one contrast is considered.

See Also

[chord_dat](#)

Examples

```
## Not run:
#Load the included dataset
data(EC)

#Generating the binary matrix
chord<-chord_dat(circ,EC$genes,EC$process)

#Creating the chord plot
GOChord(chord)

#Creating the chord plot genes ordered by logFC and a different logFC color scale
GOChord(chord,space=0.02,gene.order='logFC',lfc.col=c('red','black','cyan'))

## End(Not run)
```

GOCircle

Circular visualization of the results of a functional analysis.

Description

The circular plot combines gene expression and gene-annotation enrichment data. A subset of terms is displayed like the GOBar plot in combination with a scatterplot of the gene expression data. The whole plot is drawn on a specific coordinate system to achieve the circular layout. The segments are labeled with the term ID.

Usage

```
GOCircle(data, title, nsub, rad1, rad2, table.legend = T, zsc.col, lfc.col,
  label.size, label.fontface)
```

Arguments

<code>data</code>	A special data frame which should be the result of <code>circle_dat</code>
<code>title</code>	The title of the plot
<code>nsub</code>	A numeric or character vector. If it's numeric then the number defines how many processes are displayed (starting from the first row of data). If it's a character string of processes then these processes are displayed
<code>rad1</code>	The radius of the inner circle (default=2)
<code>rad2</code>	The radius of the outer circle (default=3)
<code>table.legend</code>	Shall a table be displayd or not? (default=TRUE)
<code>zsc.col</code>	Character vector to define the color scale for the z-score of the form <code>c(high, midpoint,low)</code>
<code>lfc.col</code>	A character vector specifying the color for up- and down-regulated genes
<code>label.size</code>	Size of the segment labels (default=5)
<code>label.fontface</code>	Font style of the segment labels (default='bold')

Details

The outer circle shows a scatter plot for each term of the logFC of the assigned genes. The colors can be changed with the argument `lfc.col`.

The `nsub` argument needs a bit more explanation to be used wisely. First of all, it can be a numeric or a character vector. If it is a character vector then it contains the IDs or term descriptions of the displayed processes. If `nsub` is a numeric vector then the number defines how many terms are displayed. It starts with the first row of the input data frame.

See Also

[circle_dat](#), [GOBar](#)

Examples

```
## Not run:
#Load the included dataset
data(EC)

#Building the circ object
circ<-circular_dat(EC$david, EC$genelist)

#Creating the circular plot
GOCircle(circ)

#Creating the circular plot with a different color scale
GOCircle(circ,zsc.col=c('yellow','black','cyan'))

#Creating the circular plot with different font style
GOCircle(circ,label.size=5,label.fontface='italic')

## End(Not run)
```

 GOCluster

Circular dendrogram.

Description

GOCluster generates a circular dendrogram of the data clustering using by default euclidean distance and average linkage. The inner ring displays the color coded logFC while the outside one encodes the assigned terms to each gene.

Usage

```
GOCluster(data, process, metric, clust, clust.by, nlfc, lfc.col, lfc.min,
          lfc.max, lfc.space, lfc.width, term.col, term.space, term.width)
```

Arguments

data	A data frame which should be the result of <code>circle_dat</code> in case the data contains only one logFC column. Otherwise data is a data frame whereas the first column contains the genes, the second the term and the following columns the logFCs of the different contrasts.
process	A character vector of selected processes (ID or term description)
metric	A character vector specifying the distance measure to be used (default='euclidean'), see <code>dist</code>
clust	A character vector specifying the agglomeration method to be used (default='average'), see <code>hclust</code>
clust.by	A character vector specifying if the clustering should be done for gene expression pattern or functional categories. By default the clustering is done based on the functional categories.
nlfc	If TRUE data contains multiple logFC columns (default= FALSE)
lfc.col	Character vector to define the color scale for the logFC of the form <code>c(high, mid-point,low)</code>
lfc.min	Specifies the minimum value of the logFC scale (default = -3)
lfc.max	Specifies the maximum value of the logFC scale (default = 3)
lfc.space	The space between the leafs of the dendrogram and the ring for the logFC
lfc.width	The width of the logFC ring
term.col	A character vector specifying the colors of the term bands
term.space	The space between the logFC ring and the term ring
term.width	The width of the term ring

Details

The `clust.by` argument keenly affects the layout of the resulting plot. The choice depends on the biological question that should be answered by creating this plot. If the purpose of the plot is to display the relationship between genes of the same functional terms `clust.by` should be set to "term". The more common option is to cluster the genes based on their expression profile by setting `clust.by` to "logFC".

Another argument which needs a bit more explanation is `n1fc`. Differential expression analysis can be performed for multiple conditions and/or batches. Therefore, the data frame contains more than one logFC value per gene. To adjust to this situation the `n1fc` argument is used. It is a numeric value and it defines the number of logFC columns within the binary membership matrix. The default is "1" assuming that most of the time only one contrast is considered.

Examples

```
## Not run:
#Load the included dataset
data(EC)

#Generating the circ object
circ<-circular_dat(EC$david, EC$genelist)

#Creating the cluster plot
GOcluster(circ, EC$process)

#Cluster the data according to gene expression and assigning a different color scale for the logFC
GOcluster(circ,EC$process,clust.by='logFC',lfc.col=c('darkgoldenrod1','black','cyan1'))

## End(Not run)
```

GOVenn

Venn diagram of differentially expressed genes.

Description

The function compares lists of differentially expressed genes and illustrates possible relations. Additionally it represents the variety of gene expression patterns within the intersection in small pie charts with three segments. Clockwise are shown the number of commonly up-regulated, commonly down-regulated and contra-regulated genes.

Usage

```
GOVenn(data1, data2, data3, title, label, lfc.col, circle.col, plot = T)
```

Arguments

<code>data1</code>	A data frame consisting of two columns: ID, logFC
<code>data2</code>	A data frame consisting of two columns: ID, logFC

data3	A data frame consisting of two columns: ID, logFC
title	The title of the plot
label	A character vector to define the legend keys
lfc.col	A character vector determining the background colors of the pie segments representing up- and down- regulated genes
circle.col	A character vector to assign clockwise colors for the circles
plot	If TRUE only the venn diagram is plotted. Otherwise the function returns a list with two items: the actual plot and a list containing the overlap entries (default=TRUE)

Details

The plot argument can be used to adjust the amount of information that is returned by calling the function. If you are only interested in the actual plot of the venn diagram, plot should be set to TRUE. Sometimes you also want to know the elements of the intersections. In this case plot should be set to FALSE and the function call will return a list of two items. The first item, that can be accessed by \$plot, contains the plotting information. Additionally, a list (\$table) will be returned containing the elements of the various overlaps.

Examples

```
## Not run:
#Load the included dataset
data(EC)

#Generating the circ object
circ<-circular_dat(EC$david, EC$genelist)

#Selecting terms of interest
l1<-subset(circ,term=='heart development',c(genes,logFC))
l2<-subset(circ,term=='plasma membrane',c(genes,logFC))
l3<-subset(circ,term=='tissue morphogenesis',c(genes,logFC))

GOVenn(l1,l2,l3, label=c('heart development','plasma membrane','tissue morphogenesis'))

## End(Not run)
```

theme_blank

theme blank

Description

theme blank

Usage

theme_blank

Format

```
List of 11
 $ axis.line      : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.text.x    : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.text.y    : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.ticks     : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.title.x   : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.title.y   : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ panel.background: list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ panel.border   : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ panel.grid.major: list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ panel.grid.minor: list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ plot.background : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 - attr(*, "class")= chr [1:2] "theme" "gg"
 - attr(*, "complete")= logi FALSE
```

Index

*Topic **datasets**

EC, [5](#)

theme_blank, [14](#)

bezier, [2](#)

chord_dat, [2](#), [9](#), [10](#)

circle_dat, [4](#), [8](#), [11](#), [12](#)

circleFun, [3](#)

draw_table, [5](#)

EC, [5](#)

get_overlap, [6](#)

get_overlap2, [6](#)

GOBar, [7](#), [11](#)

GOBubble, [8](#)

GOChord, [2](#), [3](#), [9](#)

GOCircle, [10](#)

GOCluster, [12](#)

GOVenn, [13](#)

theme_blank, [14](#)