Package 'GOfan'

November 12, 2026

Type Package

Title Sunburst Plot for Enriched Gene Ontology Terms

Version 1.0.0

Description GOfan provides an intuitive and compact visualization of Gene Ontology (GO) enrichment results using a sunburst layout inspired by SynGO, preserving hierarchical relationships among GO terms and allowing color-based encoding of information such as p-values or gene counts. By converting complex GO DAGs into clean, circular representations, it allows researchers to quickly grasp the hierarchical structure and biological significance of enriched terms. The interactive and customizable visualizations facilitate exploration of key GO categories, enhancing interpretation and presentation of enrichment analyses.

LazyData false

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

biocViews Visualization, GO, Annotation

Depends R (>= 4.5.0), ggplot2

Imports AnnotationDbi, grid, grDevices, GO.db, igraph, methods, plotly, rlang, stats, scales, vctrs

Suggests BiocStyle, knitr, rmarkdown, testthat, org.Dr.eg.db

URL https://github.com/jianhong/GOfan

BugReports https://github.com/jianhong/GOfan/issues

git_url https://git.bioconductor.org/packages/GOfan

git_branch RELEASE_3_23

git_last_commit 26f6395

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Jianhong Ou [aut, cre] (ORCID: <https://orcid.org/0000-0002-8652-2488>), Kenneth Poss [aut, fnd]

Maintainer Jianhong Ou <jou@morgridge.org>

Contents

GOfan-package 2
filterGraph 3
geom_sunburst 4
getAncestors 7
getGOalias 8
getGraph 9
ggSunburst 9
goGraph 11
simplifyDAG 11
sunburstGO 12

Index **14**

`GOfan-package` *Sunburst Plot for Enriched Gene Ontology Terms*

Description

GOfan provides a simple and intuitive way to visualize Gene Ontology (GO) enrichment results using a sunburst layout inspired by SynGO. Unlike graph-based or dot plot methods, it preserves hierarchical relationships among GO terms while maintaining a clean, interpretable view. GOfan accepts any enrichment results containing GO identifiers and uses color to represent additional information such as p-values or gene counts, making GO analysis both informative and visually engaging.

Author(s)

Jianhong Ou

Maintainer: Jianhong Ou jianhong.ou@morgridge.org

See Also

Useful links:

- https://github.com/jianhong/GOfan
- Report bugs at https://github.com/jianhong/GOfan/issues

`filterGraph` *Cut the enriched GO terms by the distances from the root after simplify*

Description

Cut the input igraph object by the distances from the root.

Usage

```
filterGraph(
  g,
  leaveTerms,
  cutoff = 4,
  filterNodesByEdgeNumber = 0,
  mustkeep = c(),
  onlyKeep = c()
)
```

Arguments

`g`	A igraph object
`leaveTerms`	Leaves must contained GO terms.
`cutoff`	The cutoff distance from the root
`filterNodesByEdgeNumber`	Filter the graphs by the edge number.
`mustkeep`	The GO terms must be kept.
`onlyKeep`	Only keep branches with give GO terms.

Value

A igraph object after filtering

Examples

```
library(igraph)
g_gnp <- sample_gnp(n = 25, p = 0.05)
filterGraph(g_gnp, leaveTerms = V(g_gnp), cutoff = 2)
```

geom_sunburst *Sunburst plot*

Description

create a sunburst geom.

Usage

```
geom_sunburst(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  lineend = "butt",
  linejoin = "mitre",
  parse = FALSE,
  size.unit = "mm",
  check_overlap = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot2::ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See ggplot2::fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
stat	The statistical transformation to use on the data for this layer. When using a geom_*() function to construct a layer, the stat argument can be used to override the default coupling between geoms and stats. The stat argument accepts the following: • A Stat ggproto subclass, for example StatCount.

- A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`.
- For more information and other ways to specify the stat, see the layer stat documentation.

`position` A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following:

- The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position.
- A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`.
- For more information and other ways to specify the position, see the layer position documentation.

`...` Other arguments passed on to `ggplot2::layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `ggplot2::layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`lineend` Line end style (round, butt, square).

`linejoin` Line join style (round, mitre, bevel).

`parse` If `TRUE`, the labels will be parsed into expressions and displayed as described in `?plotmath`.

`size.unit` How the `size` aesthetic is interpreted: as millimetres (`"mm"`, default), points (`"pt"`), centimetres (`"cm"`), inches (`"in"`), or picas (`"pc"`).

check_overlap	If TRUE, text that overlaps previous text in the same layer will not be plotted. check_overlap happens at draw time and in the order of the data. Therefore data should be arranged by the label column before calling geom_text(). Note that this argument is not supported by geom_label().
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. ggplot2::annotation_borders().

Details

Please note that the width and height aesthetics are not true position aesthetics and therefore are not subject to scale transformation. It is only after transformation that these aesthetics are applied.

Value

A ggplot object

Aesthetics

geom_sunburst understands the following aesthetics:

- x define the locations of x.
- y define the locations of y.
- xmin define the bottom of rectangle
- ymin define the left of rectangle
- xmax define the top of rectangle
- ymax define the right of rectangle
- width define the width of rectangle
- height define the height of rectangle
- colour rectangle border color
- fill rectangle fill color
- alpha rectangle fill alpha and font alpha
- linewidth line width for rectangle
- linetype line type for rectangle
- label label text
- angle label angle
- family label family

- `size` label default size
- `fontcolour` font color
- `fontface` font face
- `lineheight` font line height
- `hjust` horizontal just for label
- `vjust` vertical just for label
- `rotate90` Rotate the labels 90 degree or not. Default NULL will try to auto rotate the labels according the space.
- `sub_rect` A proportional sub-rectangle representing a share of the whole.

Examples

```
plotdata <- data.frame(
    id = c("GO:0023052", "GO:0007267", "GO:0099536", "GO:0099537", "GO:0098916"),
    x = 0.5,
    y = seq.int(5),
    xmin = 0,
    ymin = c(0.5, 1.5, 2.5, 3.5, 4.5),
    xmax = 1,
    ymax = c(1.5, 2.5, 3.5, 4.5, 5.5),
    fill = seq(1, 5),
    label = c(
        "signaling", "cell-cell signaling", "synaptic signaling",
        "trans-synaptic signaling", "anterograde trans-synaptic signaling"
    )
)
library(ggplot2)
ggplot(plotdata, aes(
    x = x, y = y, xmin = xmin, ymin = ymin, xmax = xmax, ymax = ymax,
    fill = fill, label = label
)) +
    geom_sunburst(size = 0.5, angle = -90) +
    coord_polar()
```

---

getAncestors *Recursive function to extract (ancestor, offspring) pairs*

Description

Recursive function to extract (ancestor, offspring) pairs

Usage

```
getAncestors(GO_IDs, onto = c("BP", "CC", "MF"))
```

Arguments

GO_IDs	The Gene ontology term ids
onto	The category of the GO ids. It should be one of "BP", "CC", or "MF".

Value

A data frame with columns ancestor and offspring

Examples

```
ids <- c(
    "GO:0099536", "GO:0099537", "GO:0007268",
    "GO:0098916", "GO:0050804", "GO:0099177"
)
df <- getAncestors(ids, onto = "BP")
head(df)
```

---

getGOalias *Extract all the entrez IDs from a given GO IDs*

Description

Extract all the entrez IDs from a given GO IDs

Usage

```
getGOalias(GO_IDs, org)
```

Arguments

GO_IDs	The Gene ontology term id
org	The OrgDb

Value

A list of entrez IDs for the given GO IDs

Examples

```
library(org.Dr.eg.db)
ids <- c(
    "GO:0099536", "GO:0099537", "GO:0007268",
    "GO:0098916", "GO:0050804", "GO:0099177"
)
eids <- getGOalias(ids, org.Dr.eg.db)
```

getGraph *Prepare the graph for Sunburst plot*

Description

By a given GO enrichment results, extract GO ancestor and offspring info from GO.db and then generate a simplified tree like graph.

Usage

```
getGraph(df, org, termID = "ID", onto = c("BP", "CC", "MF"))
```

Arguments

df	A data frame with enriched GO terms
org	An Go3AnnDbBimap object eg org.Dr.egGO2ALLEGS
termID	Column name in df which store the GO IDs
onto	The ontology category of the GO IDs

Value

A igraph graph.

Examples

```
library(org.Dr.eg.db)
goids <- c("GO:0099536", "GO:0099537", "GO:0007268", "GO:0098916", "GO:0050804")
g <- getGraph(data.frame(ID = goids), org = org.Dr.eg.db, onto = "BP")
```

ggSunburst *Creates sunburst diagram using ggplot2.*

Description

Creates sunburst diagram using ggplot2.

Usage

```
ggSunburst(
  plotdata,
  fontsize = 1,
  rotate90 = NULL,
  maxCharacters = 30,
  legendTitle = "color",
  start = 0,
  end = NULL,
  clip = "off",
  expand = FALSE,
  ...
)
```

Arguments

plotdata	A data.frame.
fontsize	Default fontsize.
rotate90	Rotate the labels 90 degree or not. Default NULL will try to auto rotate the labels according the space.
maxCharacters	Maximal number of characters for labels
legendTitle	The title of the legend.
start, end	Offset of starting or ending point from 12 o'clock in radians. see coord_radial.
clip	Should drawing be clipped to the extent of the plot panel? Default "on" means yes.
expand	If TRUE, adds a small expansion factor the the limits to prevent overlap between data and axes. If FALSE, the default, limits are taken directly from the scale.
...	Other parameters (except theta) passed to coord_radial.

Value

A ggplot object

Examples

```
plotdata <- data.frame(
    id = c("GO:0023052", "GO:0007267", "GO:0099536", "GO:0099537", "GO:0098916"),
    x = 0.5,
    y = seq.int(5),
    xmin = 0,
    ymin = c(0.5, 1.5, 2.5, 3.5, 4.5),
    xmax = 1,
    ymax = c(1.5, 2.5, 3.5, 4.5, 5.5),
    fill = seq(1, 5),
    label = c(
        "signaling", "cell-cell signaling", "synaptic signaling",
        "trans-synaptic signaling", "anterograde trans-synaptic signaling"
```

```
    )
)
ggSunburst(plotdata, end = pi / 2)
```

---

goGraph *Creating igraph graphs from ancestor_offspring data frame*

Description

This function creates an igraph graph from one data frames containing the ancestor and offspring information.

Usage

```
goGraph(df)
```

Arguments

df	A data frame, output of getAncestors

Value

A igraph graph.

Examples

```
goids <- c("GO:0099536", "GO:0099537", "GO:0007268", "GO:0098916", "GO:0050804")
anc <- getAncestors(goids, onto = "BP")
goGraph(anc)
```

---

simplifyDAG *Simplify DAG by keeping strongest parent in sub-graphs*

Description

simplify graph by keeping only strongest parent, to make the DAG to a tree like graph

Usage

```
simplifyDAG(g, org)
```

Arguments

g	A igraph object
org	A Go3AnnDbBimap object

Value

Simplified graph

Examples

```
library(igraph)
library(org.Dr.eg.db)
edges <- data.frame(
    ancestor = c("GO:0007154", "GO:0007267", "GO:0099536", "GO:0099537"),
    offspring = c("GO:0099536", "GO:0099536", "GO:0099537", "GO:0098916")
)
g <- graph_from_data_frame(edges)
g1 <- simplifyDAG(g, org.Dr.eg.db)
```

---

sunburstGO *Sunburst plot for enriched GO term*

Description

Sunburst plot for enriched GO term

Usage

```
sunburstGO(
  df,
  org,
  g,
  termID = "ID",
  fill = "qvalue",
  sub_rect = NULL,
  GO_annotation_level_cutoff = 4,
  filterNodesByEdgeNumber = 2,
  mustkeep = c(),
  onlyKeep = c(),
  fillNAby0 = TRUE,
  onto = c("BP", "CC", "MF"),
  plotBy = c("plotly", "ggplot2"),
  ...
)
```

Arguments

df	A data frame with enriched GO terms
org	An OrgDb object
g	An igraph graph. Output of getGraph.
termID	Column name in df which store the GO IDs

fill	Column name in df used to set the fill colors
sub_rect	Column name in df used to set the area of a proportional sub-rectangle, which represent a share of the whole. The values should be a number in the range from 0 to 1. If it is a count number, it will be convert to a proportion by divided the total number of features in the term. Otherwise, will simply re-scale to .
GO_annotation_level_cutoff	The cutoff of the GO annotation levels
filterNodesByEdgeNumber	Filter the sub graphs by the edge numbers.
mustkeep	The GO terms must be kept.
onlyKeep	Only keep branches with give GO terms.
fillNAby0	Fill the NA values by 0 or not for the color column.
onto	The ontology category of the GO IDs
plotBy	plot tools, plotly or ggplot2.
...	parameter passed to ggSunburst.

Value

A plot handle

Examples

```
library(org.Dr.eg.db)
df <- data.frame(
    ID = c("GO:0007267", "GO:0099536", "GO:0099537", "GO:0098916"),
    qvalue = -10 * log10(runif(4, max = 0.05))
)
sunburstGO(df, org.Dr.eg.db)
```

Index

∗ **package**
GOfan-package, 2

coord_radial, *10*

filterGraph, 3

geom_sunburst, 4
getAncestors, 7, *11*
getGOalias, 8
getGraph, 9, *12*
ggplot, *6*, *10*
ggplot2::aes(), *4*
ggplot2::annotation_borders(), *6*
ggplot2::fortify(), *4*
ggplot2::ggplot(), *4*
ggplot2::layer(), *5*
ggSunburst, 9, *13*
GOfan *(*GOfan-package*)*, 2
GOfan-package, 2
goGraph, 11

key glyphs, *5*

layer position, *5*
layer stat, *5*

simplifyDAG, 11
sunburstGO, 12