Package 'UpSetR'

July 21, 2025

Title A More Scalable Alternative to Venn and Euler Diagrams for Visualizing Intersecting Sets

Version 1.4.0

Date 2019-5-9

URL http://github.com/hms-dbmi/UpSetR

BugReports http://github.com/hms-dbmi/UpSetR/issues

Description Creates visualizations of intersecting sets using a novel matrix design, along with visualizations of several common set, element and attribute related tasks (Conway 2017) <doi:10.1093/bioinformatics/btx364>.

Depends R (>= 3.0)

Imports ggplot2, gridExtra, plyr, utils, stats, methods, grDevices, scales

License MIT + file LICENSE

LazyData true

VignetteBuilder knitr

Suggests knitr

RoxygenNote 6.1.0

NeedsCompilation no

Author Jake Conway [cre], Nils Gehlenborg [aut]

Maintainer Jake Conway <jake_conway@hms.harvard.edu>

Repository CRAN

Date/Publication 2019-05-22 23:30:03 UTC

Contents

`elements` *Element query for queries parameter*

Description

A query paramter to visualize specific elements of interest if queries = active (in custom plots only)

Usage

```
elements(func, query, ...)
```

Arguments

`func`	A functions provided internally
`query`	Input from the queries parameter assigned to the element function.
`...`	Additional parameters to be supplied internally

Note

See examples section of upset function on how to use this function in the queries parameter.

`fromExpression` *Expression to UpSetR converters*

Description

A function to convert an expression to a data frame compatible with UpSetR.

Usage

```
fromExpression(input)
```

Arguments

`input`	An vector (expression) to be converted to an input compatible with UpSetR

Note

See "Basic Usage" vignette for an example on how to use this function in UpSetR.

fromList — *List of named vectors to UpSetR converter*

Description

A function to convert a list of named vectors to a data frame compatible with UpSetR.

Usage

```
fromList(input)
```

Arguments

input	A list of named vectors to be converted to a data frame compatible with UpSetR

Note

See "Basic Usage" vignette for an example on how to use this function in UpSetR.

histogram — *Histogram for custom plot*

Description

A pre-made histogram that can be added to custom.plot parameter.

Usage

```
histogram(mydata, x)
```

Arguments

mydata	A data set containing intersection data provided internally
x	The x aesthetic of for the histogram plot

Note

See examples section for upset function on how to use custom.plot parameter

intersects *Intersection query for queries parameter*

Description

A query paramter to visualize elements contained in specific intersections

Usage

```
intersects(func, query, ...)
```

Arguments

func	A functions provided internally
query	Input from the queries parameter assigned to the intersection function.
...	Additional parameters to be applied internally

Note

See examples section of upset function on how to use this function in the queries parameter.

scatter_plot *Scatterplot for customplot*

Description

A pre-made scatter plot that can be added to the custom.plot parameter.

Usage

```
scatter_plot(mydata, x, y)
```

Arguments

mydata	A data set containing intersection data provided internally
x	The x aesthetic for the scatter plot
y	The y aesthetic for the scatter plot

Note

See examples section for upset function on how to use custom.plot parameter.

upset *UpSetR Plot*

Description

Visualization of set intersections using novel UpSet matrix design.

Usage

```
upset(data, nsets = 5, nintersects = 40, sets = NULL,
  keep.order = F, set.metadata = NULL, intersections = NULL,
  matrix.color = "gray23", main.bar.color = "gray23",
  mainbar.y.label = "Intersection Size", mainbar.y.max = NULL,
  sets.bar.color = "gray23", sets.x.label = "Set Size",
  point.size = 2.2, line.size = 0.7, mb.ratio = c(0.7, 0.3),
  expression = NULL, att.pos = NULL, att.color = main.bar.color,
  order.by = c("freq", "degree"), decreasing = c(T, F),
  show.numbers = "yes", number.angles = 0, group.by = "degree",
  cutoff = NULL, queries = NULL, query.legend = "none",
  shade.color = "gray88", shade.alpha = 0.25, matrix.dot.alpha = 0.5,
  empty.intersections = NULL, color.pal = 1, boxplot.summary = NULL,
  attribute.plots = NULL, scale.intersections = "identity",
  scale.sets = "identity", text.scale = 1, set_size.angles = 0,
  set_size.show = FALSE, set_size.numbers_size = NULL,
  set_size.scale_max = NULL)
```

Arguments

data	Data set
nsets	Number of sets to look at
nintersects	Number of intersections to plot. If set to NA, all intersections will be plotted.
sets	Specific sets to look at (Include as combinations. Ex: c("Name1", "Name2"))
keep.order	Keep sets in the order entered using the sets parameter. The default is FALSE, which orders the sets by their sizes.
set.metadata	Metadata that offers insight to an attribute of the sets. Input should be a data frame where the first column is set names, and the remaining columns are attributes of those sets. To learn how to use this parameter it is highly suggested to view the set metadata vignette. The link can be found on the package's GitHub page.
intersections	Specific intersections to include in plot entered as a list of lists. Ex: list(list("Set name1", "Set name2"), list("Set name1", "Set name3")). If data is entered into this parameter the only data shown on the UpSet plot will be the specific intersections listed.
matrix.color	Color of the intersection points
main.bar.color	Color of the main bar plot

`mainbar.y.label`
: The y-axis label of the intersection size bar plot

`mainbar.y.max`
: The maximum y value of the intersection size bar plot scale. May be useful when aligning multiple UpSet plots horizontally.

`sets.bar.color`
: Color of set size bar plot

`sets.x.label`
: The x-axis label of the set size bar plot

`point.size`
: Size of points in matrix plot

`line.size`
: Width of lines in matrix plot

`mb.ratio`
: Ratio between matrix plot and main bar plot (Keep in terms of hundredths)

`expression`
: Expression to subset attributes of intersection or element query data. Enter as string (Ex: "ColName > 3")

`att.pos`
: Position of attribute plot. If NULL or "bottom" the plot will be at below UpSet plot. If "top" it will be above UpSet plot

`att.color`
: Color of attribute histogram bins or scatterplot points for unqueried data represented by main bars. Default set to color of main bars.

`order.by`
: How the intersections in the matrix should be ordered by. Options include frequency (entered as "freq"), degree, or both in any order.

`decreasing`
: How the variables in order.by should be ordered. "freq" is decreasing (greatest to least) and "degree" is increasing (least to greatest)

`show.numbers`
: Show numbers of intersection sizes above bars

`number.angles`
: The angle of the numbers atop the intersection size bars

`group.by`
: How the data should be grouped ("degree" or "sets")

`cutoff`
: The number of intersections from each set (to cut off at) when aggregating by sets

`queries`
: Unified query of intersections, elements, and custom row functions. Entered as a list that contains a list of queries. query is the type of query being conducted. params are the parameters of the query (if any). color is the color of the points on the plot that will represent the query. If no color is selected one will be provided automatically. active takes TRUE or FALSE, and if TRUE, it will overlay the bars present with the results from the query. If FALSE a tick mark will indicate the intersection size. See examples section on how to do this.

`query.legend`
: Position query legend on top or bottom of UpSet plot

`shade.color`
: Color of row shading in matrix

`shade.alpha`
: Transparency of shading in matrix

`matrix.dot.alpha`
: Transparency of the empty intersections points in the matrix

`empty.intersections`
: Additionally display empty sets up to nintersects

`color.pal`
: Color palette for attribute plots

`boxplot.summary`
: Boxplots representing the distribution of a selected attribute for each intersection. Select attributes by entering a character vector of attribute names (e.g. c("Name1", "Name2")). The maximum number of attributes that can be entered is 2.

`attribute.plots`
: Create custom ggplot using intersection data represented in the main bar plot. Prior to adding custom plots, the UpSet plot is set up in a 100 by 100 grid. The attribute.plots parameter takes a list that contains the number of rows that should be allocated for the custom plot, and a list of plots with specified positions. nrows is the number of rows the custom plots should take up. There is already 100 allocated for the custom plot. plots takes a list that contains a function that returns a custom ggplot and the x and y aesthetics for the function. ncols is the number of columns that your ggplots should take up. See examples for how to add custom ggplots.

`scale.intersections`
: The scale to be used for the intersection sizes. Options: "identity", "log10", "log2"

`scale.sets`
: The scale to be used for the set sizes. Options: "identity", "log10", "log2"

`text.scale`
: Numeric, value to scale the text sizes, applies to all axis labels, tick labels, and numbers above bar plot. Can be a universal scale, or a vector containing individual scales in the following format: c(intersection size title, intersection size tick labels, set size title, set size tick labels, set names, numbers above bars)

`set_size.angles`
: Numeric, angle to rotate the set size plot x-axis text

`set_size.show`
: Logical, display the set sizes on the set size bar chart

`set_size.numbers_size`
: If set_size.show is TRUE, adjust the size of the numbers

`set_size.scale_max`
: Increase the maximum of set size scale

Details

Visualization of set data in the layout described by Lex and Gehlenborg in http://www.nature.com/nmeth/journal/v11/n8/abs/nmeth.3033.html. UpSet also allows for visualization of queries on intersections and elements, along with custom queries queries implemented using Hadley Wickham's apply function. To further analyze the data contained in the intersections, the user may select additional attribute plots to be displayed alongside the UpSet plot. The user also has the the ability to pass their own plots into the function to further analyze data belonging to queries of interest. Most aspects of the UpSet plot are customizable, allowing the user to select the plot that best suits their style. Depending on how the features are selected, UpSet can display between 25-65 sets and between 40-100 intersections.

Note

Data set must be formatted as described on the original UpSet github page: http://github.com/VCG/upset/wiki.

References

Lex et al. (2014). UpSet: Visualization of Intersecting Sets IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis 2014), vol 20, pp. 1983-1992, (2014). http://people.seas.harvard.edu/~alex/papers/2014_infovis_upset.pdf

Lex and Gehlenborg (2014). Points of view: Sets and intersections. Nature Methods 11, 779 (2014). http://www.nature.com/nmeth/journal/v11/n8/abs/nmeth.3033.html

See Also

Original UpSet Website: http://vcg.github.io/upset/about/

UpSetR github for additional examples: http://github.com/hms-dbmi/UpSetR

Examples

```
movies <- read.csv( system.file("extdata", "movies.csv", package = "UpSetR"), header=TRUE, sep=";" )

require(ggplot2); require(plyr); require(gridExtra); require(grid);

between <- function(row, min, max){
  newData <- (row["ReleaseDate"] < max) & (row["ReleaseDate"] > min)
}

plot1 <- function(mydata, x){
  myplot <- (ggplot(mydata, aes_string(x= x, fill = "color"))
            + geom_histogram() + scale_fill_identity()
            + theme(plot.margin = unit(c(0,0,0,0), "cm")))
}

plot2 <- function(mydata, x, y){
  myplot <- (ggplot(data = mydata, aes_string(x=x, y=y, colour = "color"), alpha = 0.5)
            + geom_point() + scale_color_identity()
            + theme_bw() + theme(plot.margin = unit(c(0,0,0,0), "cm")))
}

attributeplots <- list(gridrows = 55,
                  plots = list(list(plot = plot1, x= "ReleaseDate",  queries = FALSE),
                         list(plot = plot1, x= "ReleaseDate", queries = TRUE),
                   list(plot = plot2, x = "ReleaseDate", y = "AvgRating", queries = FALSE),
                   list(plot = plot2, x = "ReleaseDate", y = "AvgRating", queries = TRUE)),
                    ncols = 3)

upset(movies, nsets = 7, nintersects = 30, mb.ratio = c(0.5, 0.5),
      order.by = c("freq", "degree"), decreasing = c(TRUE,FALSE))

upset(movies, sets = c("Drama", "Comedy", "Action", "Thriller", "Western", "Documentary"),
      queries = list(list(query = intersects, params = list("Drama", "Action")),
           list(query = between, params = list(1970, 1980), color = "red", active = TRUE)))

upset(movies, attribute.plots = attributeplots,
     queries = list(list(query = between, params = list(1920, 1940)),
                    list(query = intersects, params = list("Drama"), color= "red"),
                 list(query = elements, params = list("ReleaseDate", 1990, 1991, 1992))),
      main.bar.color = "yellow")
```

Index