Package ‘vennplot’

July 22, 2025

Type Package

Title Venn Diagrams in 2D and 3D

Version 1.0

Date 2017-04-20

Description Calculate and plot Venn diagrams in 2D and 3D.

License GPL (>= 3)

Imports Rcpp (>= 0.12.7), stringr, rgl, stats, grDevices, graphics, utils

LinkingTo Rcpp

LazyData true

RoxygenNote 6.0.1

NeedsCompilation yes

Author Zehao Xu [aut, cre],
R. Wayne Oldford [aut],
Martin Lysy [ctb]

Maintainer Zehao Xu <z267xu@uwaterloo.ca>

Repository CRAN

Date/Publication 2017-10-17 20:54:02 UTC

Contents

`sharks` *Data on human encounters with great white sharks.*

Description

Data on human encounters with great white sharks.

Usage

```
sharks
```

Format

A dataset with 65 rows and 11 columns.

Year Years encounter sharks

Sex Sex of victims

Age Age of victims

Time Encounter sharks in AM or PM

Australia Encounter in Australia

USA Encounter in the United States

Surfing Surfing incident

Scuba Scuba-diving incident

Fatality Whether or not there was a fatality

Injury Whether or not there was an injury

Length The length of great white sharks

Source

`http://sharkattackinfo.com/shark_attack_news_sas.html`. Data collected by Professor Pierre-Jerome Bergeron, University of Ottawa.

Examples

```
vennplot(disjoint.combinations = sharks, vars = c("Au","USA","Fa","Ti"))
```

`vennplot` *Draw Venn and Euler diagram in 2D or 3D*

Description

Draw Venn and Euler diagram in 2D or 3D

Usage

```
vennplot(disjoint.combinations = NULL, vars = NULL, Delta = 0.1,
  ThreeD = FALSE, lambda = NULL, stressWay = c("sum", "combine"),
  delta = 0.01, weight = NULL, expand = NULL, twoWayGenerate = FALSE,
  scaleSearch = c("NelderMead", "lineSearch", "goldenSectionSearch", "BFGS",
  "CG", "L-BFGS-B", "SANN", "Brent"), twoWaySearch = c("lineSearch",
  "NelderMead", "goldenSectionSearch", "BFGS", "CG", "L-BFGS-B", "SANN",
  "Brent"), scaleSeachTolerance = list(value = 1e-05, proportional = FALSE),
  distanceTolerance = list(value = 1e-05, proportional = FALSE),
  lossTolerance = list(ToleranceofLoss = 1e-10, maximumStep = 10, ALPHA =
  0.01, ToleranceofStepsize = 1e-05, proportional = FALSE),
  stressBound = 0.001, maximumStep = 50, planeSize = 50, lower = -Inf,
  upper = Inf, control = list(), hessian = FALSE, mar = rep(1, 4),
  cols = NULL, alpha = 0.3, smooth = FALSE, ...)
```

Arguments

`disjoint.combinations`	Named numeric vector or data.frame where each column should be factor. See Details.
`vars`	Extract specific variables of data.frame as `disjoint.combinations`. If `vars = NULL`, all the information of data.frame will be extracted.
`Delta`	The length of step for method "lineSearch" or the initial interval of test points for method "NelderMead".
`ThreeD`	Draw Venn diagram in 3D. See Examples.
`lambda`	It can be `NULL` or a numeric vector. If `lambda = NULL`, the loss function optimize lambda, else, based on the given lambda, loss function will calculate stress respectively then return the minimum one and corresponding lambdas.
`stressWay`	If data set can be separated into a few groups, there will be two ways to express stress: one is to sum up all the stress (named "sum"; default), the other is to use total TSS divide by total RSS (named "combine").
`delta`	Closeness between groups.
`weight`	The weight of `disjoint.combinations`. It should have the same length with `disjoint.combinations`.
`expand`	If some balls should not intersect and the code fails to detect it. It is possible to be fixed manually but sacrificing stress.

`twoWayGenerate`	Boolean factor, if false, any missing intersections are set as zero.
`scaleSearch`	Provide multiple methods to optimize scale lambda. The default method is "NelderMead". See Details.
`twoWaySearch`	If two way intersections are missing, multiple methods are available to generate two way intersections. The default method is "lineSearch". See Details.
`scaleSeachTolerance`	A list with tolerance value and boolean factor " proportional". The loop of NelderMead and lineSearch in scaleSearch will end when the difference or proportional difference matches the tolerance value.
`distanceTolerance`	A list with tolerance value and boolean factor " proportional". The Newton method of finding distance will end when the difference or proportional difference matches the tolerance value.
`lossTolerance`	A list with ToleranceofLoss, maximumStep, ALPHA, ToleranceofStepsize and boolean factor "proportional". If ALPHA is null, the step size will be searched through Newton method and it will stop when step reaches the maximum step or the difference matches ToleranceofStepsize; else step size will be fixed with ALPHA . The loss will end when the difference or proportional difference or the total loss value matches the "ToleranceofLoss".
`stressBound`	The loop of method NelderMead will stop when stress is beyond the stressBound.
`maximumStep`	The maximum searching step for method NelderMead and Newton method of calculating distance.
`planeSize`	The plane size of calculating disjoint intersections numerically.
`lower`	The lower bound of the interval to be searched for the "goldenSectionSearch" and "L-BFGS-B". See Details.
`upper`	The upper bound of the interval to be searched for the "goldenSectionSearch" and "Brent". See Details.
`control`	A list of control parameters. See Details
`hessian`	Logical. A numerically differentiated Hessian matrix be returned or not. See Details.
`mar`	Plot margins.
`cols`	Color of balls. If `NULL`, rainbow color will be set.
`alpha`	Color darkness.
`smooth`	For 3D plot, if true, the balls will be much more smoother. However, based on the high resolution, if the number of balls is too much, when rotating, the new window stumbles.
`...`	Any further graphical parameters to be passed to the `plot` function.

Details

1. One way sets must be given in `disjoint.combination`. `e.g.disjoint.combination = c( B=2, AB=0.5)` is not allowed. `disjoint.combination = c(A = 0, B=2, AB=0.5)` works. 2. Except "NelderMead" and "lineSearch", "goldenSectionSearch" in `scaleSearch` and `twoWaySearch` is based on `optimize` and the rest methods are based on `optim`. 3. `lower`, `upper`, `control` and `hessian` share the same parameters with `optim`, and `lower`, `upper` can also be used in `optimize`

Value

An object of the class `vennplot` with following components:

xy centres of the balls (columns are (x, y) or (x, y, z) coordinates).

radius radii of the balls.

loss total loss of `vennplot`.

stress stress value for solution.

Author(s)

Zehao Xu and Wayne Oldford

Examples

```
# 3D Venn plot with arbitray sets
disjoint.combinations = c(A=80, B=50,C=100, D = 100,E = 100,
                         "A&C"=30, "A&D"= 30,"B&E" = 30, "A&E" = 40, h = 40, "B&h" = 10)
ve = vennplot(disjoint.combinations, ThreeD = TRUE)

# data frame
vennplot(disjoint.combinations = sharks, vars = c("Au","USA","Fa","Sex"),
         scaleSearch = "lineSearch", expand = 1.1)
```

Index