Package 'vennplot'

April 21, 2017

Type Package

Title Venn Diagrams in 2D and 3D

Version 0.9.02

Date 2017-04-20

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

License GPL (>= 3)

Imports Rcpp (>= 0.12.7), stringr, rgl

LinkingTo Rcpp

LazyData true

RoxygenNote 6.0.1

NeedsCompilation yes

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

Maintainer Zehao Xu <z267xu@uwaterloo.ca>

Repository CRAN

Date/Publication 2017-04-21 06:05:09 UTC

R topics documented:

vennplot-package *Calculate and plot Venn diagrams in 2D and 3D.*

Description

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

Examples

```
# arbitrary intersection sizes
combinations = c(A=1.8, B=0.9,C=1.3, D = 1.3,E = 1.6, AC=0.3,
                 AD= 0.3,BE = 0.3, AE = 0.4, f = 0.7,g =0.8,
                 h = 0.5, gf = 0.2, Bh = 0.1,i = 1,j = 0.4,
                 k=0.7,l = 1.4,kl = 0.2,m = 0.5,lm = 0.2,
                 o = 0.8,p = 0.9, op = 0.3)
ve = vennplot(combinations)

# binary dataset
combinations = sharks[,c(1,3:5,8)]
vennplot(combinations = combinations)
```

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

Description

Data on human encounters with great white sharks.

Usage

```
sharks
```

Format

A dataset of logicals with 65 rows and 9 columns. Each entry classifies whether or not the encounter (row) was of a given type (column):

Male Male shark

Female Female shark

AM Morning encounter

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

Source

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

Examples

```
combinations = sharks[,c(1,3:5,8)]
vennplot(combinations = combinations)
```

---

vennplot *Draw Venn diagram in 2D or 3D*

Description

Draw Venn diagram in 2D or 3D

Usage

```
vennplot(combinations = NULL, fit = 0.5, GRAM = T, main = NULL,
  arbitrary = NULL, mu = 2, NAME = NULL, ALPHA = 10^(-5),
  alpha.col = 0.3, disjoint = F, ThreeD = FALSE, delta = 0.01,
  mar = rep(1, 4), priority = 2, weight = rep(1, length(priority)))
```

Arguments

combinations	Named numeric vector or data.frame where each row is a logical vector indicating set membership. See Details.
fit	Scaling of semi-diameters; a scalar between 0.5 and 1.
GRAM	If TRUE, use the centralized GRAM matrix as the initial location. Otherwise use the center of an arbitrary circle.
main	Title of 2D plot.
arbitrary	When GRAM = FALSE, an integer designating which circle to use for inital location, or NULL to use a random circle. Argument is ignored when GRAM = TRUE.
mu	Generates defaults for unspecificed two-way intersections. See Details.
NAME	The name of each circle. See Examples.
ALPHA	Size of steepest descent.
alpha.col	Color darkness.
disjoint	If TRUE plots only the disjoint part of overlapping circles.
ThreeD	Draw Venn diagram in 3D. See Examples.
delta	Closeness betwen connected components.
mar	Plot margins.
priority	Scalar or vector specifying the fitting priority for intersections. See Examples.
weight	Priority weights. Must the the same length as priority. See Examples.

Details

The names of data sets (circles) supplied to `combination` must be single letters, e.g., `combination = c(A=1, B=2, AB=0.5)` Here `A` means the whole data-set, so does `B`, which means `AB` can be no larger than `min(A,B)`.

If a few two way intersections are unspecified, `mu` gives one way to generate them. For example, Suppose input combinations are `c(a=1,b=2,c=1,abc = 0.2)`. Then default values for the two-way interserctions are `ab = mu^(3-2)*abc = 0.2mu^(3-2)`, and `bc = mu^(3-2)*abc = 0.2mu^(3-2), ac = mu^(3-2)*abc =`

Value

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

center centers of the circles (columns are (`x`, `y`) or (`x`, `y`, `z`) coordinates).

semidiameters semi-diameters of the circles.

LOSS total loss of `vennplot`.

weighted.least.square Given specific priorities and weights, the weighted least square between input and lay-out.

Examples

```
# arbitray sets
combinations = c(A=1.8, B=0.9,C=1.3, D = 1.3,E = 1.6, AC=0.3,
                 AD= 0.3,BE = 0.3, AE = 0.4, f = 0.7,g =0.8,
                 h = 0.5, gf = 0.2, Bh = 0.1,i = 1,j = 0.4,
                 k=0.7,l = 1.4,kl = 0.2,m = 0.5,lm = 0.2,
                 o = 0.8,p = 0.9, op = 0.3)
ve = vennplot(combinations)

# named sets
# combinations = c(A=1, B=1,C=1, ABC = 0.1)
# ve = vennplot(combinations,NAME = c("Loon","Goose","Duck"))

# effect of parameter mu
# combinations = c(A=1, B=1, C=1, D=1, E=1, ABCDE = 0.1)
# par(mfrow = c(1,3))
# ve = vennplot(combinations)
# ve = vennplot(combinations,mu=2)
# ve = vennplot(combinations,mu=1.2)

# 3D Venn plot
combinations = c(A=803, B=304,C=1015, D = 1100,E = 1005,f = 967,H=3020,
                 CD = 1000,BC = 248,ABC = 185,ADE = 327,CDfH = 846,
                 I=800, J=760,K=1000, L = 1100,M = 900, IK=333,
                 JM = 251,IL= 289, KM = 412,JL = 213)
timestart <- Sys.time()
ve = vennplot(combinations, disjoint = TRUE,ThreeD = TRUE)
timeend <- Sys.time()
runningtime <- timeend-timestart
print(runningtime)

# effect of parameters priority and weight
```

```
# combinations = c(A = 79,B=29,C=58,AB = 25,AC = 44, BC=18,ABC = 1)
# par(mfrow = c(2,2))
# vennplot(combinations, priority = 2)
# vennplot(combinations, priority = 3)
# vennplot(combinations, priority = c(2,3), weight = c(30,1000))
# vennplot(combinations, priority = c(2,3), weight = c(30,10))

# binary data
# combinations = sharks[,c(1,3:5,8)]
# vennplot(combinations = combinations)
```

Index