Package 'VennDiagram'

April 18, 2016

Version 1.6.17

Type Package

Title Generate High-Resolution Venn and Euler Plots

Date 2016-04-16

Author Hanbo Chen

Maintainer Paul Boutros <Paul.Boutros@oicr.on.ca>

Depends R (>= 2.14.1), grid (>= 2.14.1), futile.logger

Description A set of functions to generate high-resolution Venn and Euler plots. Includes handling for several special cases, including two-case scaling, and extensive customization of plot shape and structure.

License GPL-2

LazyLoad yes

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2016-04-18 08:45:40

R topics documented:

VennDiagram-package *Venn diagram plotting*

Description

Functions to plot high-resolution and highly-customizable Venn and Euler plots.

Details

Package:	VennDiagram
Type:	Package
Version:	1.6.0
Date:	2013-04-10
License:	GPL-2
LazyLoad:	yes

Author(s)

Author: Hanbo Chen <Hanbert.Chen@mail.utoronto.ca>\ Maintainer: Dr. Paul C. Boutros <Paul.Boutros@utoronto.ca>

calculate.overlap *Calculate Overlap*

Description

Determine the groupings of values as they would be presented in the venn diagram.

Usage

```
calculate.overlap(x)
```

Arguments

x	A list of vectors (e.g., integers, chars), with each component corresponding to a separate circle in the Venn diagram

Details

This function mostly complements the venn.diagram() function for the case where users want to know what values are grouped into the particular areas of the venn diagram.

Value

Returns a list of lists which contain the values assigned to each of the areas of a venn diagram.

Author(s)

Christopher Lalansingh

Examples

```
# A simple single-set diagram
cardiome <- letters[1:10]
superset <- letters[8:24]
overlap <- calculate.overlap(
x = list(
"Cardiome" = cardiome,
"SuperSet" = superset
)
);
```

`draw.pairwise.venn` *Draw a Venn Diagram with Two Sets*

Description

Creates a Venn diagram with two sets. Creates Euler diagrams when the dataset meets certain conditions.

Usage

```
draw.pairwise.venn(area1, area2, cross.area, category = rep("", 2),
    euler.d = TRUE, scaled = TRUE, inverted = FALSE,
    ext.text = TRUE, ext.percent = rep(0.05, 3), lwd =
    rep(2, 2), lty = rep("solid", 2), col = rep("black",
    2), fill = NULL, alpha = rep(0.5, 2), label.col =
    rep("black", 3), cex = rep(1, 3), fontface =
    rep("plain", 3), fontfamily = rep("serif", 3), cat.pos
    = c(-50, 50), cat.dist = rep(0.025, 2), cat.cex =
    rep(1, 2), cat.col = rep("black", 2), cat.fontface =
    rep("plain", 2), cat.fontfamily = rep("serif", 2),
    cat.just = rep(list(c(0.5, 0.5)), 2), cat.default.pos
    = "outer", cat.prompts = FALSE, ext.pos = rep(0, 2),
    ext.dist = rep(0, 2), ext.line.lty = "solid",
    ext.length = rep(0.95, 2), ext.line.lwd = 1,
    rotation.degree = 0, rotation.centre = c(0.5, 0.5),
    ind = TRUE, sep.dist = 0.05, offset = 0, cex.prop =
    NULL, print.mode = "raw", sigdigs = 3, ...)
```

Arguments

area1	The size of the first set
area2	The size of the second set
cross.area	The size of the intersection between the sets
category	A vector (length 2) of strings giving the category names of the sets
euler.d	Boolean indicating whether to draw Euler diagrams when conditions are met or not (Venn Diagrams with moveable circles)
scaled	Boolean indicating whether to scale circle sizes in the diagram according to set sizes or not (euler.d must be true to enable this)
inverted	Boolean indicating whether the diagram should be mirrored long the vertical axis or not
ext.text	Boolean indicating whether to place area labels outside the circles in case of small partial areas or not
ext.percent	A vector (length 3) indicating the proportion that a partial area has to be smaller than to trigger external text placement. The elements allow for individual control of the areas in the order of area1, area2 and intersect area.
lwd	A vector (length 2) of numbers giving the line width of the circles' circumferences
lty	A vector (length 2) giving the line dash pattern of the circles' circumferences
col	A vector (length 2) giving the colours of the circles' circumferences
fill	A vector (length 2) giving the colours of the circles' areas
alpha	A vector (length 2) giving the alpha transparency of the circles' areas
label.col	A vector (length 3) giving the colours of the areas' labels
cex	A vector (length 3) giving the size of the areas' labels
fontface	A vector (length 3) giving the fontface of the areas' labels
fontfamily	A vector (length 3) giving the fontfamily of the areas' labels
cat.pos	A vector (length 2) giving the positions (in degrees) of the category names along the circles, with 0 (default) at the 12 o'clock location
cat.dist	A vector (length 2) giving the distances (in npc units) of the category names from the edges of the circles (can be negative)
cat.cex	A vector (length 2) giving the size of the category names
cat.col	A vector (length 2) giving the colours of the category names
cat.fontface	A vector (length 2) giving the fontface of the category names
cat.fontfamily	A vector (length 2) giving the fontfamily of the category names
cat.just	List of 2 vectors of length 2 indicating horizontal and vertical justification of each category name
cat.default.pos	One of c('outer', 'text') to specify the default location of category names (cat.pos and cat.dist are handled differently)

cat.prompts	Boolean indicating whether to display help text on category name positioning or not)
ext.pos	A vector (length 1 or 2) giving the positions (in degrees) of the external area labels along the circles, with 0 (default) at 12 o'clock
ext.dist	A vector (length 1 or 2) giving how far to place the external area labels relative to its anchor point
ext.line.lty	A vector (length 1 or 2) giving the dash pattern of the lines connecting the external area labels to their anchor points
ext.length	A vector (length 1 or 2) giving the proportion of the lines connecting the external area labels to their anchor points actually drawn
ext.line.lwd	A vector (length 1 or 2) giving the width of the lines connecting the external area labels to their anchor points
rotation.degree	Number of degrees to rotate the entire diagram
rotation.centre	A vector (length 2) indicating (x,y) of the rotation centre
ind	Boolean indicating whether the function is to automatically draw the diagram before returning the gList object or not
sep.dist	Number giving the distance between circles in case of an Euler diagram showing mutually exclusive sets
offset	Number between 0 and 1 giving the amount of offset from the centre in case of an Euler diagram showing inclusive sets
cex.prop	A function or string used to rescale areas
print.mode	Can be either 'raw' or 'percent'. This is the format that the numbers will be printed in. Can pass in a vector with the second element being printed under the first
sigdigs	If one of the elements in print.mode is 'percent', then this is how many significant digits will be kept
...	Additional arguments to be passed, including margin, which indicates amount of whitespace around the final diagram in npc units

Details

Euler diagrams are drawn for mutually exclusive sets (cross.area == 0), inclusive sets (area1 == 0 or area2 == 0), and coincidental sets (area1 == 0 and area2 == 0) if euler.d == TRUE. The function defaults to placing the larger set on the left. inverted or rotation.degree can be used to reverse this.

Value

Returns an object of class gList containing the grid objects that make up the diagram. Also displays the diagram in a graphical device unless specified with ind = FALSE. Grid::grid.draw can be used to draw the gList object in a graphical device.

Author(s)

Hanbo Chen

Examples

```
# A simple two-set diagram
venn.plot <- draw.pairwise.venn(100, 70, 30, c("First", "Second"));
grid.draw(venn.plot);
grid.newpage();

# Same diagram as above, but without scaling
venn.plot <- draw.pairwise.venn(100, 70, 30, c("First", "Second"), scaled = FALSE);
grid.draw(venn.plot);
grid.newpage();

# A more complicated diagram Demonstrating external area labels
venn.plot <- draw.pairwise.venn(
area1 = 100,
area2 = 70,
cross.area = 68,
category = c("First", "Second"),
fill = c("blue", "red"),
lty = "blank",
cex = 2,
cat.cex = 2,
cat.pos = c(285, 105),
cat.dist = 0.09,
cat.just = list(c(-1, -1), c(1, 1)),
ext.pos = 30,
ext.dist = -0.05,
ext.length = 0.85,
ext.line.lwd = 2,
ext.line.lty = "dashed"
);
grid.draw(venn.plot);
grid.newpage();

# Demonstrating an Euler diagram
venn.plot <- draw.pairwise.venn(
area1 = 100,
area2 = 70,
cross.area = 0,
category = c("First", "Second"),
cat.pos = c(0, 180),
euler.d = TRUE,
sep.dist = 0.03,
rotation.degree = 45
);

# Writing to file
tiff(filename = "Pairwise_Venn_diagram.tiff", compression = "lzw");
grid.draw(venn.plot);
```

```
dev.off();
```

---

draw.quad.venn *Draw a Venn Diagram with Four Sets*

Description

Creates a Venn diagram with four sets.

Usage

```
draw.quad.venn(area1, area2, area3, area4, n12, n13, n14, n23, n24,
    n34, n123, n124, n134, n234, n1234, category = rep("",
    4), lwd = rep(2, 4), lty = rep("solid", 4), col =
    rep("black", 4), fill = NULL, alpha = rep(0.5, 4),
    label.col = rep("black", 15), cex = rep(1, 15),
    fontface = rep("plain", 15), fontfamily = rep("serif",
    15), cat.pos = c(-15, 15, 0, 0), cat.dist = c(0.22,
    0.22, 0.11, 0.11), cat.col = rep("black", 4), cat.cex
    = rep(1, 4), cat.fontface = rep("plain", 4),
    cat.fontfamily = rep("serif", 4), cat.just =
    rep(list(c(0.5, 0.5)), 4), rotation.degree = 0,
    rotation.centre = c(0.5, 0.5), ind = TRUE, cex.prop =
    NULL, print.mode = "raw", sigdigs = 3, direct.area =
    FALSE, area.vector = 0, ...)
```

Arguments

area1	The size of the first set
area2	The size of the second set
area3	The size of the third set
area4	The size of the fourth set
n12	The size of the intersection between the first and the second set
n13	The size of the intersection between the first and the third set
n14	The size of the intersection between the first and the fourth set
n23	The size of the intersection between the second and the third set
n24	The size of the intersection between the second and the fourth set
n34	The size of the intersection between the third and the fourth set
n123	The size of the intersection between the first, second and third sets
n124	The size of the intersection between the first, second and fourth sets
n134	The size of the intersection between the first, third and fourth sets
n234	The size of the intersection between the second, third and fourth sets
n1234	The size of the intersection between all four sets

category	A vector (length 4) of strings giving the category names of the sets
lwd	A vector (length 4) of numbers giving the line width of the circles' circumferences
lty	A vector (length 4) giving the dash pattern of the circles' circumferences
col	A vector (length 4) giving the colours of the circles' circumferences
fill	A vector (length 4) giving the colours of the circles' areas
alpha	A vector (length 4) giving the alpha transparency of the circles' areas
label.col	A vector (length 15) giving the colours of the areas' labels
cex	A vector (length 15) giving the size of the areas' labels
fontface	A vector (length 15) giving the fontface of the areas' labels
fontfamily	A vector (length 15) giving the fontfamily of the areas' labels
cat.pos	A vector (length 4) giving the positions (in degrees) of the category names along the circles, with 0 (default) at 12 o'clock
cat.dist	A vector (length 4) giving the distances (in npc units) of the category names from the edges of the circles (can be negative)
cat.cex	A vector (length 4) giving the size of the category names
cat.col	A vector (length 4) giving the colours of the category names
cat.fontface	A vector (length 4) giving the fontface of the category names
cat.fontfamily	A vector (length 4) giving the fontfamily of the category names
cat.just	List of 4 vectors of length 2 indicating horizontal and vertical justification of each category name
rotation.degree	Number of degrees to rotate the entire diagram
rotation.centre	A vector (length 2) indicating (x,y) of the rotation centre
ind	Boolean indicating whether the function is to automatically draw the diagram before returning the gList object or not
cex.prop	A function or string used to rescale areas
print.mode	Can be either 'raw' or 'percent'. This is the format that the numbers will be printed in. Can pass in a vector with the second element being printed under the first
sigdigs	If one of the elements in print.mode is 'percent', then this is how many significant digits will be kept
direct.area	If this is equal to true, then the vector passed into area.vector will be directly assigned to the areas of the corresponding regions. Only use this if you know which positions in the vector correspond to which regions in the diagram
area.vector	An argument to be used when direct.area is true. These are the areas of the corresponding regions in the Venn Diagram
...	Additional arguments to be passed, including margin, which indicates amount of whitespace around the final diagram in npc units

Details

The function defaults to placing the ellipses so that area1 corresponds to lower left, area2 corresponds to lower right, area3 corresponds to middle left and area4 corresponds to middle right. Refer to the example below to see how the 31 partial areas are ordered. Arguments with length of 15 (label.col, cex, fontface, fontfamily) will follow the order in the example.

Value

Returns an object of class gList containing the grid objects that make up the diagram. Also displays the diagram in a graphical device unless specified with ind = FALSE. Grid::grid.draw can be used to draw the gList object in a graphical device.

Author(s)

Hanbo Chen

Examples

```
# Reference four-set diagram
venn.plot <- draw.quad.venn(
area1 = 72,
area2 = 86,
area3 = 50,
area4 = 52,
n12 = 44,
n13 = 27,
n14 = 32,
n23 = 38,
n24 = 32,
n34 = 20,
n123 = 18,
n124 = 17,
n134 = 11,
n234 = 13,
n1234 = 6,
category = c("First", "Second", "Third", "Fourth"),
fill = c("orange", "red", "green", "blue"),
lty = "dashed",
cex = 2,
cat.cex = 2,
cat.col = c("orange", "red", "green", "blue")
);

# Writing to file
tiff(filename = "Quad_Venn_diagram.tiff", compression = "lzw");
grid.draw(venn.plot);
dev.off();
```

draw.quintuple.venn *Draw a Venn Diagram with Five Sets*

Description

Creates a Venn diagram with five sets.

Usage

```
draw.quintuple.venn(area1, area2, area3, area4, area5, n12, n13, n14, n15,
    n23, n24, n25, n34, n35, n45, n123, n124, n125, n134,
    n135, n145, n234, n235, n245, n345, n1234, n1235,
    n1245, n1345, n2345, n12345, category = rep("", 5),
    lwd = rep(2, 5), lty = rep("solid", 5), col =
    rep("black", 5), fill = NULL, alpha = rep(0.5, 5),
    label.col = rep("black", 31), cex = rep(1, 31),
    fontface = rep("plain", 31), fontfamily = rep("serif",
    31), cat.pos = c(0, 287.5, 215, 145, 70), cat.dist =
    rep(0.2, 5), cat.col = rep("black", 5), cat.cex =
    rep(1, 5), cat.fontface = rep("plain", 5),
    cat.fontfamily = rep("serif", 5), cat.just =
    rep(list(c(0.5, 0.5)), 5), rotation.degree = 0,
    rotation.centre = c(0.5, 0.5), ind = TRUE, cex.prop =
    NULL, print.mode = "raw", sigdigs = 3, direct.area =
    FALSE, area.vector = 0, ...)
```

Arguments

area1	The size of the first set
area2	The size of the second set
area3	The size of the third set
area4	The size of the fourth set
area5	The size of the fifth set
n12	The size of the intersection between the first and the second set
n13	The size of the intersection between the first and the third set
n14	The size of the intersection between the first and the fourth set
n15	The size of the intersection between the first and the fifth set
n23	The size of the intersection between the second and the third set
n24	The size of the intersection between the second and the fourth set
n25	The size of the intersection between the second and the fifth set
n34	The size of the intersection between the third and the fourth set
n35	The size of the intersection between the third and the fifth set
n45	The size of the intersection between the fourth and the fifth set

n123	The size of the intersection between the first, second and third sets
n124	The size of the intersection between the first, second and fourth sets
n125	The size of the intersection between the first, second and fifth sets
n134	The size of the intersection between the first, third and fourth sets
n135	The size of the intersection between the first, third and fifth sets
n145	The size of the intersection between the first, fourth and fifth sets
n234	The size of the intersection between the second, third and fourth sets
n235	The size of the intersection between the second, third and fifth sets
n245	The size of the intersection between the second, fourth and fifth sets
n345	The size of the intersection between the third, fourth and fifth sets
n1234	The size of the intersection between the first, second, third and fourth sets
n1235	The size of the intersection between the first, second, third and fifth sets
n1245	The size of the intersection between the first, second, fourth and fifth sets
n1345	The size of the intersection between the first, third, fourth and fifth sets
n2345	The size of the intersection between the second, third, fourth and fifth sets
n12345	The size of the intersection between all five sets
category	A vector (length 5) of strings giving the category names of the sets
lwd	A vector (length 5) of numbers giving the line width of the circles' circumferences
lty	A vector (length 5) giving the dash pattern of the circles' circumferences
col	A vector (length 5) giving the colours of the circles' circumferences
fill	A vector (length 5) giving the colours of the circles' areas
alpha	A vector (length 5) giving the alpha transparency of the circles' areas
label.col	A vector (length 31) giving the colours of the areas' labels
cex	A vector (length 31) giving the size of the areas' labels
fontface	A vector (length 31) giving the fontface of the areas' labels
fontfamily	A vector (length 31) giving the fontfamily of the areas' labels
cat.pos	A vector (length 5) giving the positions (in degrees) of the category names along the circles, with 0 (default) at 12 o'clock
cat.dist	A vector (length 5) giving the distances (in npc units) of the category names from the edges of the circles (can be negative)
cat.cex	A vector (length 5) giving the size of the category names
cat.col	A vector (length 5) giving the colours of the category names
cat.fontface	A vector (length 5) giving the fontface of the category names
cat.fontfamily	A vector (length 5) giving the fontfamily of the category names
cat.just	List of 5 vectors of length 2 indicating horizontal and vertical justification of each category name
rotation.degree	Number of degrees to rotate the entire diagram

`rotation.centre`
: A vector (length 2) indicating (x,y) of the rotation centre

`ind`
: Boolean indicating whether the function is to automatically draw the diagram before returning the gList object or not

`cex.prop`
: A function or string used to rescale areas

`print.mode`
: Can be either 'raw' or 'percent'. This is the format that the numbers will be printed in. Can pass in a vector with the second element being printed under the first

`sigdigs`
: If one of the elements in print.mode is 'percent', then this is how many significant digits will be kept

`direct.area`
: If this is equal to true, then the vector passed into area.vector will be directly assigned to the areas of the corresponding regions. Only use this if you know which positions in the vector correspond to which regions in the diagram

`area.vector`
: An argument to be used when direct.area is true. These are the areas of the corresponding regions in the Venn Diagram

`...`
: Additional arguments to be passed, including `margin`, which indicates amount of whitespace around the final diagram in npc units

Details

The function defaults to placing the ellipses representing the areas 1 to 5 in a counterclockwise fashion. Refer to the example below to see how the 31 partial areas are ordered. Arguments with length of 31 (label.col, cex, fontface, fontfamily) will follow the order in the example.

Value

Returns an object of class gList containing the grid objects that make up the diagram. Also displays the diagram in a graphical device unless specified with ind = FALSE. Grid::grid.draw can be used to draw the gList object in a graphical device.

Author(s)

Hanbo Chen

Examples

```
# Reference five-set diagram
venn.plot <- draw.quintuple.venn(
area1 = 301,
area2 = 321,
area3 = 311,
area4 = 321,
area5 = 301,
n12 = 188,
n13 = 191,
n14 = 184,
n15 = 177,
n23 = 194,
```

```
n24 = 197,
n25 = 190,
n34 = 190,
n35 = 173,
n45 = 186,
n123 = 112,
n124 = 108,
n125 = 108,
n134 = 111,
n135 = 104,
n145 = 104,
n234 = 111,
n235 = 107,
n245 = 110,
n345 = 100,
n1234 = 61,
n1235 = 60,
n1245 = 59,
n1345 = 58,
n2345 = 57,
n12345 = 31,
category = c("A", "B", "C", "D", "E"),
fill = c("dodgerblue", "goldenrod1", "darkorange1", "seagreen3", "orchid3"),
cat.col = c("dodgerblue", "goldenrod1", "darkorange1", "seagreen3", "orchid3"),
cat.cex = 2,
margin = 0.05,
cex = c(1.5, 1.5, 1.5, 1.5, 1.5, 1, 0.8, 1, 0.8, 1, 0.8, 1, 0.8, 1, 0.8,
1, 0.55, 1, 0.55, 1, 0.55, 1, 0.55, 1, 0.55, 1, 1, 1, 1, 1, 1.5),
ind = TRUE
);

# Writing to file
tiff(filename = "Quintuple_Venn_diagram.tiff", compression = "lzw");
grid.draw(venn.plot);
dev.off();
```

`draw.single.venn` *Draw a Venn Diagram with a Single Set*

Description

Creates a Venn diagram with a single set.

Usage

```
draw.single.venn(area, category = "", lwd = 2, lty = "solid", col = "black", fill = NULL,
alpha = 0.5, label.col = "black", cex = 1,
fontface = "plain", fontfamily = "serif",
cat.pos = 0, cat.dist = 0.025, cat.cex = 1, cat.col = "black",
cat.fontface = "plain", cat.fontfamily = "serif",
```

```
cat.just = list(c(0.5, 0.5)),
cat.default.pos = "outer", cat.prompts = FALSE, rotation.degree = 0,
rotation.centre = c(0.5, 0.5), ind = TRUE, ...)
```

Arguments

area	The size of the set
category	The category name of the set
lwd	width of the circle's circumference
lty	dash pattern of the circle's circumference
col	Colour of the circle's circumference
fill	Colour of the circle's area
alpha	Alpha transparency of the circle's area
label.col	Colour of the area label
cex	size of the area label
fontface	fontface of the area label
fontfamily	fontfamily of the area label
cat.pos	The position (in degrees) of the category name along the circle, with 0 (default) at 12 o'clock
cat.dist	The distance (in npc units) of the category name from the edge of the circle (can be negative)
cat.cex	size of the category name
cat.col	Colour of the category name
cat.fontface	fontface of the category name
cat.fontfamily	fontfamily of the category name
cat.just	List of 1 vector of length 2 indicating horizontal and vertical justification of the category name
cat.default.pos	One of c('outer', 'text') to specify the default location of category names (cat.pos and cat.dist are handled differently)
cat.prompts	Boolean indicating whether to display help text on category name positioning or not)
rotation.degree	Number of degrees to rotate the entire diagram
rotation.centre	A vector (length 2) indicating (x,y) of the rotation centre
ind	Boolean indicating whether the function is to automatically draw the diagram in the end or not
...	Additional arguments to be passed, including margin, which indicates amount of whitespace around the final diagram in npc units

Details

This function mostly complements other functions in the VennDiagram package that draws multi-set diagrams by providing a function that draws single-set diagrams with similar graphical options.

Value

Returns an object of class gList containing the grid objects that make up the diagram. Also displays the diagram in a graphical device unless specified with ind = FALSE. Grid::grid.draw can be used to draw the gList object in a graphical device.

Author(s)

Hanbo Chen

Examples

```
# A simple single-set diagram
venn.plot <- draw.single.venn(100, "First");
grid.draw(venn.plot);
grid.newpage();

# A more complicated diagram
venn.plot <- draw.single.venn(
area = 365,
category = "All\nDays",
lwd = 5,
lty = "blank",
cex = 3,
label.col = "orange",
cat.cex = 4,
cat.pos = 180,
cat.dist = -0.20,
cat.col = "white",
fill = "red",
alpha = 0.15
);
grid.draw(venn.plot);
grid.newpage();

# Writing to file
tiff(filename = "Single_Venn_diagram.tiff", compression = "lzw");
venn.plot <- draw.single.venn(100, "First", ind = FALSE);
grid.draw(venn.plot);
dev.off();
```

`draw.triple.venn` *Draw a Venn Diagram with Three Sets*

Description

Creates a Venn diagram with three sets. Creates Euler diagrams when the dataset meets certain conditions.

Usage

```
draw.triple.venn(area1, area2, area3, n12, n23, n13, n123, category =
    rep("", 3), rotation = 1, reverse = FALSE, euler.d =
    TRUE, scaled = TRUE, lwd = rep(2, 3), lty =
    rep("solid", 3), col = rep("black", 3), fill = NULL,
    alpha = rep(0.5, 3), label.col = rep("black", 7), cex
    = rep(1, 7), fontface = rep("plain", 7), fontfamily =
    rep("serif", 7), cat.pos = c(-40, 40, 180), cat.dist =
    c(0.05, 0.05, 0.025), cat.col = rep("black", 3),
    cat.cex = rep(1, 3), cat.fontface = rep("plain", 3),
    cat.fontfamily = rep("serif", 3), cat.just =
    list(c(0.5, 1), c(0.5, 1), c(0.5, 0)), cat.default.pos
    = "outer", cat.prompts = FALSE, rotation.degree = 0,
    rotation.centre = c(0.5, 0.5), ind = TRUE, sep.dist =
    0.05, offset = 0, cex.prop = NULL, print.mode = "raw",
    sigdigs = 3, direct.area = FALSE, area.vector = 0,
    ...)
```

Arguments

`area1`	The size of the first set
`area2`	The size of the second set
`area3`	The size of the third set
`n12`	The size of the intersection between the first and the second set
`n23`	The size of the intersection between the second and the third set
`n13`	The size of the intersection between the first and the third set
`n123`	The size of the intersection between all three sets
`category`	A vector (length 3) of strings giving the category names of the sets
`rotation`	1 (default), 2, or 3 indicating clockwise rotation of the three sets from the default arrangement
`reverse`	Boolean indicating whether the diagram should be mirrored long the vertical axis or not
`euler.d`	Boolean indicating whether to draw Euler diagrams when conditions are met or not (Venn Diagrams with moveable circles)

scaled	Boolean indicating whether to scale circle sizes in certain Euler diagrams according to set sizes or not (euler.d must be true to enable this)
lwd	A vector (length 3) of numbers giving the width of the circles' circumferences
lty	A vector (length 3) giving the dash pattern of the circles' circumferences
col	A vector (length 3) giving the colours of the circles' circumferences
fill	A vector (length 3) giving the colours of the circles' areas
alpha	A vector (length 3) giving the alpha transparency of the circles' areas
label.col	A vector (length 7) giving the colours of the areas' labels
cex	A vector (length 7) giving the size of the areas' labels
fontface	A vector (length 7) giving the fontface of the areas' labels
fontfamily	A vector (length 7) giving the fontfamily of the areas' labels
cat.pos	A vector (length 3) giving the positions (in degrees) of the category names along the circles, with 0 (default) at 12 o'clock
cat.dist	A vector (length 3) giving the distances (in npc units) of the category names from the edges of the circles (can be negative)
cat.cex	A vector (length 3) giving the size of the category names
cat.col	A vector (length 3) giving the colours of the category names
cat.fontface	A vector (length 3) giving the fontface of the category names
cat.fontfamily	A vector (length 3) giving the fontfamily of the category names
cat.just	List of 3 vectors of length 2 indicating horizontal and vertical justification of each category name
cat.default.pos	One of c('outer', 'text') to specify the default location of category names (cat.pos and cat.dist are handled differently)
cat.prompts	Boolean indicating whether to display help text on category name positioning or not)
rotation.degree	Number of degrees to rotate the entire diagram
rotation.centre	A vector (length 2) indicating (x,y) of the rotation centre
ind	Boolean indicating whether the function is to automatically draw the diagram before returning the gList object or not
sep.dist	Number between 0 and 1 giving the distance between circles in certain Euler diagrams with mutually exclusive sets
offset	Number giving the amount of offset from the centre in certain Euler diagrams with inclusive sets
cex.prop	A function or string used to rescale areas
print.mode	Can be either 'raw' or 'percent'. This is the format that the numbers will be printed in. Can pass in a vector with the second element being printed under the first

sigdigs	If one of the elements in print.mode is 'percent', then this is how many significant digits will be kept
direct.area	If this is equal to true, then the vector passed into area.vector will be directly assigned to the areas of the corresponding regions. Only use this if you know which positions in the vector correspond to which regions in the diagram
area.vector	An argument to be used when direct.area is true. These are the areas of the corresponding regions in the Venn Diagram
...	Additional arguments to be passed, including margin, which indicates amount of whitespace around the final diagram in npc units

Details

Euler diagrams are drawn for 19 special cases if euler.d == TRUE. Certain Euler diagrams make use of the scaled, sep.dist, or offset arguments specific to two-set Venn diagrams where appropriate. The function defaults to placing the three circles in a triangular arrangement with two sets on top and one set below. The circles correspond to area1, area2 and area3 in a clockwise fashion with area1 on the top left. N.B. General scaling for three-set Venn diagrams are disabled due to potentially misleading visual representation of the data. To re-enable, assign any value to variable overrideTriple.

Value

Returns an object of class gList containing the grid objects that make up the diagram. Also displays the diagram in a graphical device unless specified with ind = FALSE. Grid::grid.draw can be used to draw the gList object in a graphical device.

Author(s)

Hanbo Chen

Examples

```
# A simple three-set diagram
venn.plot <- draw.triple.venn(65, 75, 85,
 35, 15, 25, 5, c("First", "Second", "Third"));
grid.draw(venn.plot);
grid.newpage();

# A more complicated diagram
venn.plot <- draw.triple.venn(
area1 = 65,
area2 = 75,
area3 = 85,
n12 = 35,
n23 = 15,
n13 = 25,
n123 = 5,
category = c("First", "Second", "Third"),
fill = c("blue", "red", "green"),
lty = "blank",
```

```
cex = 2,
cat.cex = 2,
cat.col = c("blue", "red", "green")
);
grid.draw(venn.plot);
grid.newpage();

# Demonstrating an Euler diagram
venn.plot <- draw.triple.venn(20, 40, 60, 0, 0, 0, 0,
 c("First", "Second", "Third"), sep.dist = 0.1, rotation.degree = 30);

# Writing to file
tiff(filename = "Triple_Venn_diagram.tiff", compression = "lzw");
grid.draw(venn.plot);
dev.off();
```

`get.venn.partitions` *Get the size of individual partitions in a Venn diagram*

Description

Partitions a list into Venn regions.

Usage

```
get.venn.partitions(x, force.unique = TRUE, keep.elements = TRUE,
  hierarchical = FALSE)
```

Arguments

`x`	A list of vectors.
`force.unique`	A logical value. Should only unique values be considered?
`keep.elements`	A logical value. Should the elements in each region be returned?
`hierarchical`	A logical value. Changed the way overlapping elements are treated if force.unique is TRUE.

Value

A data frame with `length(x)` columns and `2 ^ length(x)` rows. The first `length(x)` columns are all logical; see `make.truth.table` for more details. There are three additional columns:

..set.. A set theoretical desription of the Venn region. (Note that in some locales under Windows, the data.frame print method fails to correctly display the Unicode symbols for set union and set intersection. This is a bug in R, not this function.)

..values.. A vector of values contained in the Venn region. Not returned if keep.elements is FALSE.

..count.. An integer of the number of values in the Venn region.

Details

If force.unique is FALSE, then there are two supported methods of grouping categories with duplicated elements in common. If hierarchical is FALSE, then any common elements are gathered into a pool. So if `x <- list(a = c(1,1,2,2,3,3), b=c(1,2,3,4,4,5), c=c(1,4))` then `(b intersect c)/(a)` would contain three 4's. Since the 4's are pooled, `(b)/(a union c)` contains no 4's. If hierachical is TRUE, then `(b intersect c)/(a)` would contain one 4.Then `(b)/(a union c)` cotains one 4.

Author(s)

Richard Cotton.

See Also

`venn.diagram`, `make.truth.table`

Examples

```
# Compare force.unique options
x <- list(a = c(1, 1, 1, 2, 2, 3), b = c(2, 2, 2, 3, 4, 4))
get.venn.partitions(x)
get.venn.partitions(x, force.unique = FALSE)

# Figure 1D from ?venn.diagram
xFig1d = list(
  I = c(1:60, 61:105, 106:140, 141:160, 166:175, 176:180, 181:205,
       206:220),
  IV = c(531:605, 476:530, 336:375, 376:405, 181:205, 206:220, 166:175,
        176:180),
  II = c(61:105, 106:140, 181:205, 206:220, 221:285, 286:335, 336:375,
          376:405),
  III = c(406:475, 286:335, 106:140, 141:160, 166:175, 181:205, 336:375,
           476:530)
 )
get.venn.partitions(xFig1d)
grid.draw(VennDiagram::venn.diagram(x, NULL))
```

`make.truth.table` *Make a truth table*

Description

Makes a truth table of the inputs.

Usage

```
make.truth.table(x)
```

Arguments

x	A short vector.

Value

A data frame with `length(x)` logical vector columns and `2 ^ length(x)` rows.

Author(s)

Richard Cotton

See Also

`expand.grid`

Examples

```
## Not run: make.truth.table(c(a = 1, b = 2, c = 3, d = 4))
```

---

venn.diagram *Make a Venn Diagram*

Description

This function takes a list and creates a publication-quality TIFF Venn Diagram

Usage

```
venn.diagram(x, filename, height = 3000, width = 3000, resolution =
    500, imagetype = "tiff", units = "px", compression =
    "lzw", na = "stop", main = NULL, sub = NULL, main.pos
    = c(0.5, 1.05), main.fontface = "plain",
    main.fontfamily = "serif", main.col = "black",
    main.cex = 1, main.just = c(0.5, 1), sub.pos = c(0.5,
    1.05), sub.fontface = "plain", sub.fontfamily =
    "serif", sub.col = "black", sub.cex = 1, sub.just =
    c(0.5, 1), category.names = names(x), force.unique =
    TRUE, print.mode = "raw", sigdigs = 3, direct.area =
    FALSE, area.vector = 0, hyper.test = FALSE, total.population = NULL,
    lower.tail = TRUE, ...)
```

Arguments

x	A list of vectors (e.g., integers, chars), with each component corresponding to a separate circle in the Venn diagram
filename	Filename for image output, or if NULL returns the grid object itself
height	Integer giving the height of the output figure in units
width	Integer giving the width of the output figure in units
resolution	Resolution of the final figure in DPI
imagetype	Specification of the image format (e.g. tiff, png or svg)
units	Size-units to use for the final figure
compression	What compression algorithm should be applied to the final tiff
na	Missing value handling method: "none", "stop", "remove"
main	Character giving the main title of the diagram
sub	Character giving the subtitle of the diagram
main.pos	Vector of length 2 indicating (x,y) of the main title
main.fontface	Character giving the fontface (font style) of the main title
main.fontfamily	Character giving the fontfamily (font type) of the main title
main.col	Character giving the colour of the main title
main.cex	Number giving the cex (font size) of the main title
main.just	Vector of length 2 indicating horizontal and vertical justification of the main title
sub.pos	Vector of length 2 indicating (x,y) of the subtitle
sub.fontface	Character giving the fontface (font style) of the subtitle
sub.fontfamily	Character giving the fontfamily (font type) of the subtitle
sub.col	Character Colour of the subtitle
sub.cex	Number giving the cex (font size) of the subtitle
sub.just	Vector of length 2 indicating horizontal and vertical justification of the subtitle
category.names	Allow specification of category names using plotmath syntax
force.unique	Logical specifying whether to use only unique elements in each item of the input list or use all elements. Defaults to FALSE
print.mode	Can be either 'raw' or 'percent'. This is the format that the numbers will be printed in. Can pass in a vector with the second element being printed under the first
sigdigs	If one of the elements in print.mode is 'percent', then this is how many significant digits will be kept
direct.area	If this is equal to true, then the vector passed into area.vector will be directly assigned to the areas of the corresponding regions. Only use this if you know which positions in the vector correspond to which regions in the diagram
area.vector	An argument to be used when direct.area is true. These are the areas of the corresponding regions in the Venn Diagram

hyper.test	If there are only two categories in the venn diagram and total.population is not NULL, then perform the hypergeometric test and add it to the sub title.
total.population	An argument to be used when hyper.test is true. This is the total population size
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
...	A series of graphical parameters tweaking the plot. See below for details

Details

Argument	Venn Sizes	Class	Description
lwd	1,2,3,4,5	*numeric*	Vector giving the width of each circle's circumference
lty	1,2,3,4,5	*numeric*	Vector giving the dash pattern of each circle's circumference
col	1,2,3,4,5	*character*	Vector giving the colour of each circle's circumference
fill	1,2,3,4,5	*character*	Vector giving the colour of each circle's area
alpha	1,2,3,4,5	*numeric*	Vector giving the alpha transparency of each circle's area
label.col	1,2,3,4,5	*character*	Vector giving the colour for each area label (length = 1/3/7/15 based on set-nu
cex	1,2,3,4,5	*numeric*	Vector giving the size for each area label (length = 1/3/7/15 based on set-numb
fontface	1,2,3,4,5	*character*	Vector giving the fontface for each area label (length = 1/3/7/15 based on set-n
fontfamily	1,2,3,4,5	*character*	Vector giving the fontfamily for each area label (length = 1/3/7/15 based on se
cat.pos	1,2,3,4,5	*numeric*	Vector giving the position (in degrees) of each category name along the circle,
cat.dist	1,2,3,4,5	*numeric*	Vector giving the distance (in npc units) of each category name from the edge
cat.cex	1,2,3,4,5	*numeric*	Vector giving the size for each category name
cat.col	1,2,3,4,5	*character*	Vector giving the colour for each category name
cat.fontface	1,2,3,4,5	*character*	Vector giving the fontface for each category name
cat.fontfamily	1,2,3,4,5	*character*	Vector giving the fontfamily for each category name
cat.just	1,2,3,4,5	*numeric*	List (length = 1/2/3/4 based on set number) of Vectors of length 2 indicating h
cat.default.pos	1,2,3	*character*	One of c('outer', 'text') to specify the default location of category names (cat.
margin	1,2,3,4,5	*numeric*	Number giving the amount of whitespace around the diagram in grid units
rotation.degree	1,2,3,4,5	*numeric*	Number of degrees to rotate the entire diagram
rotation.centre	1,2,3,4,5	*numeric*	Vector of length 2 indicating (x,y) of the rotation centre
rotation	3	*numeric*	Number giving the clockwise rotation of a three-set Venn diagram (1, 2, or 3)
reverse	3	*logical*	Reflect the three-set Venn diagram along its central vertical axis of symmetry.
euler.d	2, 3	*logical*	Enable Euler diagrams for two-set and three-set Venn diagrams (Venn Diagra
scaled	2, 3	*logical*	Enable scaling for two-set and certain three-set Euler diagrams. (euler.d must
sep.dist	2, 3	*numeric*	Controls the separation between distinct circles in certain two-set or three-set
offset	2, 3	*numeric*	Number between 0 and 1 giving the amount to offset the smaller circle by in t
inverted	2	*logical*	Flip the two-set Venn diagram along its vertical axis (distinguished from reve
ext.text	2	*logical*	Allow external text labels when areas are small
ext.percent	2	*numeric*	A vector (length 3) indicating the proportion that a partial area has to be small
ext.line.lwd	2	*numeric*	lwd of line connecting to ext.text
ext.dist	2	*numeric*	Vector of length 1 or 2 indicating length of external line (use negative values t
ext.length	2	*numeric*	Vector of length 1 or 2 indicating the proportion of the external line that is dra

Value

Plots a figure to the file given by the *filename* argument.

Author(s)

Hanbo Chen

See Also

draw.single.venn, draw.pairwise.venn, draw.triple.venn, draw.quad.venn, draw.quintuple.venn

Examples

```
# Note: most examples are listed as dontrun to meet CRAN requirements,
# but all should work as-is!

# compact and minimal notation
## Not run:
venn.plot <- venn.diagram(
list(A = 1:150, B = 121:170),
"Venn_2set_simple.tiff"
);
venn.plot <- venn.diagram(
list(A = 1:150, B = 121:170, C = 101:200),
"Venn_3set_simple.tiff"
);

## End(Not run)

# a more elaborate two-set Venn diagram with title and subtitle
venn.plot <- venn.diagram(
x = list(
"A" = 1:100,
"B" = 96:140
),
filename = "Venn_2set_complex.tiff",
scaled = TRUE,
ext.text = TRUE,
ext.line.lwd = 2,
ext.dist = -0.15,
ext.length = 0.9,
ext.pos = -4,
inverted = TRUE,
cex = 2.5,
cat.cex = 2.5,
rotation.degree = 45,
main = "Complex Venn Diagram",
sub = "Featuring: rotation and external lines",
main.cex = 2,
sub.cex = 1
);
```

```
## Not run:
# sample three-set Euler diagram
venn.plot <- venn.diagram(
x = list(
"Num A" = paste("Num", 1:100),
"Num B" = c(paste("Num", 61:70), paste("Num", 71:100)),
"Num C" = c(paste("Num", 41:60), paste("Num", 61:70))),
euler.d = TRUE,
filename = "Euler_3set_simple.tiff",
cat.pos = c(-20, 0, 20),
cat.dist = c(0.05, 0.05, 0.02),
cex = 2.5,
cat.cex = 2.5,
reverse = TRUE
);

# sample three-set Euler diagram
venn.plot <- venn.diagram(
x = list(
A = c(1:10),
B = c(11:90),
C = c(81:90)
),
euler.d = TRUE,
filename = "Euler_3set_scaled.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = 0
);

## End(Not run)

# sample four-set Venn Diagram
A <- sample(1:1000, 400, replace = FALSE);
B <- sample(1:1000, 600, replace = FALSE);
C <- sample(1:1000, 350, replace = FALSE);
D <- sample(1:1000, 550, replace = FALSE);
E <- sample(1:1000, 375, replace = FALSE);

venn.plot <- venn.diagram(
x = list(
A = A,
D = D,
B = B,
C = C
),
filename = "Venn_4set_pretty.tiff",
col = "transparent",
fill = c("cornflowerblue", "green", "yellow", "darkorchid1"),
alpha = 0.50,
label.col = c("orange", "white", "darkorchid4", "white",
"white", "white", "white", "white", "darkblue", "white",
"white", "white", "white", "darkgreen", "white"),
```

```
cex = 1.5,
fontfamily = "serif",
fontface = "bold",
cat.col = c("darkblue", "darkgreen", "orange", "darkorchid4"),
cat.cex = 1.5,
cat.pos = 0,
cat.dist = 0.07,
cat.fontfamily = "serif",
rotation.degree = 270,
margin = 0.2
);

# sample five-set Venn Diagram
venn.plot <- venn.diagram(
x = list(
A = A,
B = B,
C = C,
D = D,
E = E
),
filename = "Venn_5set_pretty.tiff",
col = "black",
fill = c("dodgerblue", "goldenrod1", "darkorange1", "seagreen3", "orchid3"),
alpha = 0.50,
cex = c(1.5, 1.5, 1.5, 1.5, 1.5, 1, 0.8, 1, 0.8, 1, 0.8, 1, 0.8,
 1, 0.8, 1, 0.55, 1, 0.55, 1, 0.55, 1, 0.55, 1, 0.55, 1, 1, 1, 1, 1, 1.5),
cat.col = c("dodgerblue", "goldenrod1", "darkorange1", "seagreen3", "orchid3"),
cat.cex = 1.5,
cat.fontface = "bold",
margin = 0.05
);

# Complex three-way Venn with labels & sub-/super-scripts
venn.plot <- venn.diagram(
x = list(
I = c(1:60, 61:105, 106:140, 141:160, 166:175, 176:180, 181:205,
206:220),
II = c(531:605, 476:530, 336:375, 376:405, 181:205, 206:220, 166:175,
176:180),
III = c(61:105, 106:140, 181:205, 206:220, 221:285, 286:335, 336:375,
376:405)
),
category.names = c(
expression( bold('A'['1: subscript']) ),
expression( bold('B'^'2: going up') ),
expression( paste(bold('C'^'3'), bold('X'['i' <= 'r'^'2']^'2') ) )
),
filename = 'Fig3-1_triple_labels_sub_and_superscripts.tiff',
output = TRUE,
height = 3000,
width = 3000,
resolution = 300,
```

```
compression = 'lzw',
units = 'px',
lwd = 6,
lty = 'blank',
fill = c('yellow', 'purple', 'green'),
cex = 3.5,
fontface = "bold",
fontfamily = "sans",
cat.cex = 3,
cat.fontface = "bold",
cat.default.pos = "outer",
cat.pos = c(-27, 27, 135),
cat.dist = c(0.055, 0.055, 0.085),
cat.fontfamily = "sans",
rotation = 1
);

# Complex 3-way Venn using expressions
venn.plot <- venn.diagram(
   x = list(
"Num A" = paste("Num", 1:100),
"Num B" = c(paste("Num", 61:70), paste("Num", 71:100)),
"Num C" = c(paste("Num", 41:60), paste("Num", 61:70))),
category.names = c(
expression( bold('A'['1']) ),
expression( bold('A'['2']) ),
expression( bold('A'['3']) )
),
euler.d = TRUE,
filename = "Fig3-2_Euler_3set_simple_with_subscripts.tiff",
cat.pos = c(-20, 0, 20),
cat.dist = c(0.05, 0.05, 0.02),
cex = 2.5,
cat.cex = 2.5,
reverse = TRUE
);

## Not run:
# Example to print to screen
venn.plot <- venn.diagram(
x = list(
sample1 = c(1:40),
sample2 = c(30:60)
),
filename = NULL
);

# Save picture to non-TIFF file type
# currently working on adding this functionality directly into venn.diagram
venn.plot <- venn.diagram(
x = list (
A = 1:10,
B = 6:25
```

```
),
filename = NULL
);

jpeg("venn_jpeg.jpg");
grid.draw(venn.plot);
dev.off();


## End(Not run)

#dontrun-starts-here
### NB: All figures from the paper can be run, but are turned off from
###     automatic execution to reduce burden on CRAN computing resources.
## Not run:
# Figure 1A
venn.plot <- venn.diagram(
x = list(
Label = 1:100
),
filename = "1A-single_Venn.tiff",
col = "black",
lwd = 9,
fontface = "bold",
fill = "grey",
alpha = 0.75,
cex = 4,
cat.cex = 3,
cat.fontface = "bold",
);

# Figure 1B
venn.plot <- venn.diagram(
x = list(
X = 1:150,
Y = 121:180
),
filename = "1B-double_Venn.tiff",
lwd = 4,
fill = c("cornflowerblue", "darkorchid1"),
alpha = 0.75,
label.col = "white",
cex = 4,
fontfamily = "serif",
fontface = "bold",
cat.col = c("cornflowerblue", "darkorchid1"),
cat.cex = 3,
cat.fontfamily = "serif",
cat.fontface = "bold",
cat.dist = c(0.03, 0.03),
cat.pos = c(-20, 14)
);
```

```
# Figure 1C
venn.plot <- venn.diagram(
x = list(
R = c(1:70, 71:110, 111:120, 121:140),
B = c(141:200, 71:110, 111:120, 201:230),
G = c(231:280, 111:120, 121:140, 201:230)
),
filename = "1C-triple_Venn.tiff",
col = "transparent",
fill = c("red", "blue", "green"),
alpha = 0.5,
label.col = c("darkred", "white", "darkblue", "white",
 "white", "white", "darkgreen"),
cex = 2.5,
fontfamily = "serif",
fontface = "bold",
cat.default.pos = "text",
cat.col = c("darkred", "darkblue", "darkgreen"),
cat.cex = 2.5,
cat.fontfamily = "serif",
cat.dist = c(0.06, 0.06, 0.03),
cat.pos = 0
);

# Figure 1D
venn.plot <- venn.diagram(
x = list(
I = c(1:60, 61:105, 106:140, 141:160, 166:175, 176:180, 181:205,
206:220),
IV = c(531:605, 476:530, 336:375, 376:405, 181:205, 206:220, 166:175,
176:180),
II = c(61:105, 106:140, 181:205, 206:220, 221:285, 286:335, 336:375,
376:405),
III = c(406:475, 286:335, 106:140, 141:160, 166:175, 181:205, 336:375,
476:530)
),
filename = "1D-quadruple_Venn.tiff",
col = "black",
lty = "dotted",
lwd = 4,
fill = c("cornflowerblue", "green", "yellow", "darkorchid1"),
alpha = 0.50,
label.col = c("orange", "white", "darkorchid4", "white", "white", "white",
 "white", "white", "darkblue", "white",
  "white", "white", "white", "darkgreen", "white"),
cex = 2.5,
fontfamily = "serif",
fontface = "bold",
cat.col = c("darkblue", "darkgreen", "orange", "darkorchid4"),
cat.cex = 2.5,
cat.fontfamily = "serif"
);
```

```
# Figure 2-1
venn.plot <- venn.diagram(
x = list(
A = 1:105,
B = 101:115
),
filename = "2-1_special_case_ext-text.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = c(-20, 20),
ext.line.lty = "dotted",
ext.line.lwd = 2,
ext.pos = 12,
ext.dist = -0.12,
ext.length = 0.85
);

# Figure 2-2
venn.plot <- venn.diagram(
x = list(
A = 1:100,
B = 1:10
),
filename = "2-2_special_case_pairwise-inclusion.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = 0
);

# Figure 2-3
venn.plot <- venn.diagram(
x = list(
A = 1:150,
B = 151:250
),
filename = "2-3_special_case_pairwise-exclusion.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = c(0, 0),
cat.dist = 0.05
);

# Figure 2-4
venn.plot <- venn.diagram(
x = list(
A = c(1:50, 101:140, 141:160, 161:170),
B = c(171:230, 101:140, 161:170, 291:320),
C = c(141:160, 161:170, 291:320)
),
filename = "2-4_triple_special_case-001.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.dist = c(0.05, 0.05, -0.1)
```

```
);

# Figure 2-5
venn.plot <- venn.diagram(
x = list(
A = c(1:100),
B = c(61:70, 71:100),
C = c(41:60, 61:70)
),
filename = "2-5_triple_special_case-012AA.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = c(-25, 0, 30),
cat.dist = c(0.05, 0.05, 0.02)
);

# Figure 2-6
venn.plot <- venn.diagram(
x = list(
A = c(1:90),
B = c(1:25),
C = c(1:5)
),
filename = "2-6_triple_special_case-022AAAO.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = 0,
cat.dist = c(0.03, 0.03, 0.01)
);

# Figure 2-7
venn.plot <- venn.diagram(
x = list(
A = c(1:20),
B = c(21:80),
C = c(81:210)
),
filename = "2-7_triple_special_case-100.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.dist = 0.05
);

# Figure 2-8
venn.plot <- venn.diagram(
x = list(
A = c(1:80),
B = c(41:150),
C = c(71:100)
),
filename = "2-8_triple_special_case-011A.tiff",
cex = 2.5,
cat.cex = 2.5,
```

```
cat.dist = c(0.07, 0.07, 0.02),
cat.pos = c(-20, 20, 20)
);

# Figure 2-9
venn.plot <- venn.diagram(
x = list(
A = c(1:10),
B = c(11:90),
C = c(81:90)
),
filename = "2-9_triple_special_case-121AO.tiff",
cex = 2.5,
cat.cex = 2.5,
cat.pos = 0,
cat.dist = c(0.04, 0.04, 0.02),
reverse = TRUE
);

#dontrun-ends-here

## End(Not run)
```

Index