Package 'GGally'

August 26, 2014

Version 0.4.8

Date 2014-08-25

Maintainer Barret Schloerke <schloerke@gmail.com>

License GPL (>= 2.0)

Title Extension to ggplot2.

Type Package

LazyLoad yes

LazyData true

Author Barret Schloerke <schloerke@gmail.com>, Jason Crowley <crowley.jason.s@gmail.com>, Di Cook <dicook@iastate.edu>, Heike Hofmann <hofmann@iastate.edu>, Hadley Wickham <h.wickham@iastate.edu>, and Francois Briatte <f.briatte@ed.ac.uk>, Moritz Marbach <mmarbach@mail.uni-mannheim.de>, and Edwin Thoen <edwinthoen@@gmail.com>

Description GGally is designed to be a helper to ggplot2. It contains templates for different plots to be combined into a plot matrix, a parallel coordinate plot function, as well as a function for making a network plot.

Depends R (>= 2.14)

Imports ggplot2 (>= 1.0.0), grid (>= 3.0.0), gtable (>= 0.1.2), plyr (>= 1.8), reshape (>= 0.8.4), stringr (>= 0.6.2)

Suggests arm (>= 1.7), intergraph (>= 2.0-0), network (>= 1.7.2),RColorBrewer (>= 1.0-5), scales (>= 0.2.3), scagnostics (>= 0.2-4), sna (>= 2.3-1), survival (>= 2.37-4), tnet (>= 3.0),roxygen2 (>= 4.0.1)

NeedsCompilation no

Repository CRAN

Date/Publication 2014-08-26 17:37:04

R topics documented:

`cityServiceFirms` *City and service firms data from the UCIrvine Network Data Repository*

Description

This data extract is taken from UCIrvine Network Data Repository. The original description follows, with minor edits.

Usage

```
data(cityServiceFirms)
```

Format

A list with 5 values: mel, gal, val, iel, oel. Each value is a list of size 101.

Details

These data consist of the distribution of offices for 46 'global' advanced producer service firms over 55 world cities. Global firms are defined by having offices in at least 15 different cities. This data consists of two modes where firms are linked to cities.

These data consist of the distribution of offices for 46 'global' advanced producer service firms over 55 world cities. Global firms are defined by having offices in at least 15 different cities. World cities are from the GaWC inventory of world cities (see GaWC Research Bulletin 6). For each service sector three levels of presence were identified-prime, major and minor-on the basis of size and importance of offices. Thus each city can be scored 0 (not qualified) to 3 (prime center) for each sector. Sums of these scores produces a figure which indicates a city's 'world city-ness' up to a maximum of 12. See http://www.lboro.ac.uk/gawc for more info. The service level is represented via an edge attribute 'level'.

Each vertex also contain an attribute called 'type'. This attribute represent the vertex mode - whether it is a city or firm. For the firm the vertices are further categorized by their service sector. For example, Accountancy Firm, Advertising Firm, Banking and Finance Firm, and Law Firm.

These data are an experimental set of data derived from Data Set 6 (43 of the firms qualify as global) but with three additional law firms added which do not have London offices. For publications that make use of these data, http://www.lboro.ac.uk/gawc.

References

https://networkdata.ics.uci.edu/netdata/html/cities.html

P.J. Taylor and D.R.F. Walker. "World Cities and Global Firms." http://www.lboro.ac.uk/gawc/datasets/da6.html.

Christopher L. DuBois, Emma S. Spiro, Zack Almquist, Mark S. Handcock, David Hunter, Carter T. Butts, Steven M. Goodreau, and Martina Morris. 2003 netdata: A Collection of Network Data Smith, Tom W., Peter V. Marsden, Michael Hout, Jibum Kim. *General Social Surveys, 1972-2006*.

---

get.VarTypes *Get vector of variable types from data frame*

Description

Get vector of variable types from data frame

Usage

```
get.VarTypes(df)
```

Arguments

df	data frame to extract variable types from

Value

character vector with variable types, with names corresponding to the variable names from df

Author(s)

Jason Crowley <crowley.jason.s@gmail.com>

---

getPlot *getPlot*

Description

Retrieves the ggplot object at the desired location.

Usage

```
getPlot(plotMatrix, rowFromTop, columnFromLeft)
```

Arguments

plotMatrix	ggpair object to select from
rowFromTop	row from the top
columnFromLeft	column from the left

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 plotMatrix2 <- ggpairs(tips[,3:2], upper = list(combo = "denstrip"))
 getPlot(plotMatrix2, 1, 2)
```

ggally_barDiag *Plots the Bar Plots by Using Diagonal*

Description

Plots the bar plots by using Diagonal.

Usage

```
ggally_barDiag(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguements are sent to geom_bar

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(movies, package = "ggplot2")
ggally_barDiag(movies, mapping = ggplot2::aes(x = mpaa))
# ggally_barDiag(movies, mapping = ggplot2::aes_string(x = "mpaa"))
# ggally_barDiag(movies, mapping = ggplot2::aes_string(x ="rating", binwidth = ".1"))
```

ggally_blank *Blank*

Description

Draws nothing.

Usage

```
ggally_blank(...)
```

Arguments

...	other arguments ignored

Details

Makes a "blank" ggplot object that will only draw white space

Author(s)

Barret Schloerke <schloerke@gmail.com>

---

ggally_box *Plots the Box Plot*

Description

Make a box plot with a given data set

Usage

```
ggally_box(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments being supplied to geom_boxplot

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_box(tips, mapping = ggplot2::aes(x = total_bill, y = sex))
 ggally_box(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "sex"))
 ggally_box(
   tips,
   mapping        = ggplot2::aes_string(y = "total_bill", x = "sex", color = "sex"),
   outlier.colour = "red",
   outlier.shape  = 13,
   outlier.size   = 8
 )
```

ggally_cor *Correlation from the Scatter Plot*

Description

Estimate correlation from the given data.

Usage

```
ggally_cor(data, mapping, corAlignPercent = 0.6, corMethod = "pearson",
  corUse = "complete.obs", ...)
```

Arguments

data	data set using
mapping	aesthetics being used
corAlignPercent	right align position of numbers. Default is 60 percent across the horizontal
corMethod	method suppied to cor function
corUse	use supplied to cor function
...	other arguments being supplied to geom_text

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_cor(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "tip"))
 ggally_cor(
   tips,
   mapping = ggplot2::aes_string(x = "total_bill", y = "tip", size = 15, colour = "red")
 )
 ggally_cor(
   tips,
   mapping = ggplot2::aes_string(x = "total_bill", y = "tip", color = "sex"),
   size = 5
 )
```

ggally_density *Plots the Scatter Density Plot*

Description

Make a scatter density plot from a given data.

Usage

```
ggally_density(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	parameters sent to either stat_density2d or geom_density2d

Details

The aesthetic "fill" determines whether or not stat_density2d (filled) or geom_density2d (lines) is used.

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_density(tips, mapping = ggplot2::aes(x = total_bill, y = tip))
 ggally_density(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "tip"))
 ggally_density(
   tips,
   mapping = ggplot2::aes_string(x = "total_bill", y = "tip", fill = "..level..")
 )
 ggally_density(
   tips,
   mapping = ggplot2::aes_string(x = "total_bill", y = "tip", fill = "..level..")
 ) + ggplot2::scale_fill_gradient(breaks = c(0.05, 0.1,0.15,0.2))
```

ggally_densityDiag *Plots the Density Plots by Using Diagonal*

Description

Plots the density plots by using Diagonal.

Usage

```
ggally_densityDiag(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used.
...	other arguments sent to stat_density

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_densityDiag(tips, mapping = ggplot2::aes(x = total_bill))
 #data(movies)
 #ggally_densityDiag(movies, mapping = ggplot2::aes_string(x="rating"))
 #ggally_densityDiag(movies, mapping = ggplot2::aes_string(x="rating", color = "mpaa"))
```

ggally_denstrip *Plots a tile plot with facets*

Description

Make Tile Plot as densely as possible.

Usage

```
ggally_denstrip(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments being sent to stat_bin

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_denstrip(tips, mapping = ggplot2::aes(x = total_bill, y = sex))
 ggally_denstrip(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "sex"))
 ggally_denstrip(
   tips,
   mapping = ggplot2::aes_string(x = "sex", y = "tip", binwidth = "0.2")
 ) + ggplot2::scale_fill_gradient(low = "grey80", high = "black")
```

---

ggally_diagAxis *Internal Axis Labeling Plot for ggpairs*

Description

This function is used when `axisLabels == "internal"`.

Usage

```
ggally_diagAxis(data, mapping, labelSize = 5, labelXPercent = 0.5,
  labelYPercent = 0.55, labelHJust = 0.5, labelVJust = 0.5,
  gridLabelSize = 4, ...)
```

Arguments

`data`	dataset being plotted
`mapping`	aesthetics being used (x is the variable the plot will be made for)
`labelSize`	size of variable label
`labelXPercent`	percent of horizontal range
`labelYPercent`	percent of vertical range
`labelHJust`	hjust supplied to label
`labelVJust`	vjust supplied to label
`gridLabelSize`	size of grid labels
`...`	other arguments for geom_text

Author(s)

Jason Crowley <crowley.jason.s@gmail.com> and Barret Schloerke

Examples

```
data(tips, package = "reshape")
 ggally_diagAxis(tips, ggplot2::aes(x=tip))
 ggally_diagAxis(tips, ggplot2::aes(x=sex))
```

ggally_dot *Plots the Box Plot with Dot*

Description

Add jittering with the box plot

Usage

```
ggally_dot(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments being supplied to geom_jitter

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_dot(tips, mapping = ggplot2::aes(x = total_bill, y = sex))
 ggally_dot(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "sex"))
 ggally_dot(
   tips,
   mapping = ggplot2::aes_string(y = "total_bill", x = "sex", color = "sex")
 )
 ggally_dot(
   tips,
  mapping = ggplot2::aes_string(y = "total_bill", x = "sex", color = "sex", shape = "sex")
 ) + ggplot2::scale_shape(solid=FALSE)
```

ggally_dotAndBox *Plots either Box Plot or Dot Plots*

Description

Place box plots or dot plots on the graph

Usage

```
ggally_dotAndBox(data, mapping, ..., boxPlot = TRUE)
```

Arguments

data	data set using
mapping	aesthetics being used
...	parameters passed to either geom_jitter or geom_boxplot
boxPlot	boolean to decide to plot either box plots (TRUE) or dot plots (FALSE)

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_dotAndBox(
   tips,
   mapping = ggplot2::aes(x = total_bill, y = sex, color = sex),
   boxPlot = TRUE
 )
 ggally_dotAndBox(tips, mapping = ggplot2::aes(x = total_bill, y = sex, color = sex), boxPlot=FALSE)
```

---

ggally_facetbar *Plots the Bar Plots Faceted by Conditional Variable*

Description

X variables are plotted using geom_bar and faceted by the Y variable.

Usage

```
ggally_facetbar(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguements are sent to geom_bar

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_facetbar(tips, ggplot2::aes(x = sex, y = smoker, fill = time))
 ggally_facetbar(tips, ggplot2::aes(x = smoker, y = sex, fill = time))
```

ggally_facetdensity *Plots the density plots by faceting*

Description

Make density plots by displaying subsets of the data in different panels.

Usage

```
ggally_facetdensity(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments being sent to stat_density

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_facetdensity(tips, mapping = ggplot2::aes(x = total_bill, y = sex))
 ggally_facetdensity(
   tips,
   mapping = ggplot2::aes_string(y = "total_bill", x = "sex", color = "sex")
 )
```

ggally_facetdensitystrip *Plots a density plot with facets or a tile plot with facets*

Description

Make Tile Plot as densely as possible.

Usage

```
ggally_facetdensitystrip(data, mapping, ..., den_strip = FALSE)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments being sent to either stat_bin or stat_density
den_strip	boolean to deceide whether or not to plot a density strip(TRUE) or a facet density(FALSE) plot.

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
example(ggally_facetdensity)
example(ggally_denstrip)
```

---

ggally_facethist *Plots the Histograms by Faceting*

Description

Make histograms by displaying subsets of the data in different panels.

Usage

```
ggally_facethist(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	parameters sent to stat_bin()

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_facethist(tips, mapping = ggplot2::aes(x = tip, y = sex))
 ggally_facethist(tips, mapping = ggplot2::aes_string(x = "tip", y = "sex"), binwidth = 0.1)
```

ggally_points *Plots the Scatter Plot*

Description

Make a scatter plot with a given data set.

Usage

```
ggally_points(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments are sent to geom_point

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(mtcars)
ggally_points(mtcars, mapping = ggplot2::aes(x = disp, y = hp))
ggally_points(mtcars, mapping = ggplot2::aes_string(x = "disp", y = "hp"))
ggally_points(
  mtcars,
  mapping = ggplot2::aes_string(
    x     = "disp",
    y     = "hp",
    color = "as.factor(cyl)",
    size  = "gear"
  )
)
```

ggally_ratio *Plots a mosaic plots*

Description

Plots the mosaic plot by using fluctuation.

Usage

```
ggally_ratio(data)
```

Arguments

data	data set using

Details

Must send only two discrete columns in the data set.

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(movies, package = "ggplot2")
ggally_ratio(movies[,c("mpaa","Action")])
ggally_ratio(movies[,c("mpaa","Action")]) + ggplot2::coord_equal()
nummpaa <- length(levels(movies[,"mpaa"]))
numAction <- length(levels(as.factor(movies[,"Action"])))
ggally_ratio(
  movies[,c("Action","mpaa")]
) + ggplot2::theme(
  aspect.ratio = nummpaa / numAction
)
```

---

ggally_smooth *Plots the Scatter Plot with Smoothing*

Description

Add a smoothed condition mean with a given scatter plot.

Usage

```
ggally_smooth(data, mapping, ...)
```

Arguments

data	data set using
mapping	aesthetics being used
...	other arguments to add to geom_point

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
data(tips, package = "reshape")
 ggally_smooth(tips, mapping = ggplot2::aes(x = total_bill, y = tip))
 ggally_smooth(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "tip"))
 ggally_smooth(tips, mapping = ggplot2::aes_string(x = "total_bill", y = "tip", color = "sex"))
```

---

ggally_text *GGplot Text*

Description

Plot text for a plot.

Usage

```
ggally_text(label, mapping = ggplot2::aes(color = "black"), xP = 0.5,
  yP = 0.5, xrange = c(0, 1), yrange = c(0, 1), ...)
```

Arguments

label	text that you want to appear
mapping	aesthetics that don't relate to position (such as color)
xP	horizontal position percentage
yP	vertical position percentage
xrange	range of the data around it. Only nice to have if plotting in a matrix
yrange	range of the data around it. Only nice to have if plotting in a matrix
...	other arguments for geom_text

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
ggally_text("Example 1")
ggally_text("Example\nTwo", mapping = ggplot2::aes_string(size = 15, color = "red"))
```

ggcorr *ggcorr - Plot a correlation matrix with ggplot2*

Description

Function for making a correlation plot starting from a data matrix, using ggplot2. The function is directly inspired by Tian Zheng and Yu-Sung Su's arm::corrplot function. Please visit http://github.com/briatte/ggcorr for the latest development and descriptions about ggcorr.

Usage

```
ggcorr(data, method = "pairwise", palette = "RdYlGn", name = "rho",
  geom = "tile", max_size = 6, label = FALSE, label_alpha = FALSE,
  label_color = "black", label_round = 1, ...)
```

Arguments

data	a data matrix. Should contain numerical (continuous) data.
method	a character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs". Defaults to "pairwise".
palette	a ColorBrewer palette to be used for correlation coefficients. Defaults to "RdYlGn".
name	a character string for the legend that shows quintiles of correlation coefficients.
geom	the geom object to use. Accepts either tile (the default) or circle, to plot proportionally scaled circles.
max_size	the maximum size for circles, as passed to scale_size_area for proportional scaling. Defaults to 6.
label	whether to add correlation coefficients as two-digit numbers over the plot. Defaults to FALSE.
label_alpha	whether to make the correlation coefficients transparent as they come close to 0. Defaults to FALSE.
label_color	color for the correlation coefficients. Defaults to "black".
label_round	decimal rounding of the correlation coefficients. Defaults to 1.
...	other arguments supplied to geom_text for the diagonal labels. Arguments pertaining to the title or other items can be achieved through ggplot2 methods.

Author(s)

Francois Briatte <f.briatte@ed.ac.uk>

See Also

cor and corrplot

Examples

```
# Basketball statistics provided by Nathan Yau at Flowing Data.
nba <- read.csv("http://datasets.flowingdata.com/ppg2008.csv")
# Default output.
ggcorr(nba[, -1])
# Labelled output, with coefficient transparency.
ggcorr(nba[, -1],
       label = TRUE,
       label_alpha = TRUE,
       name = "") +
  ggplot2::theme(legend.position = "bottom")
# Custom options.
ggcorr(
  nba[, -1],
  geom = "circle",
  max_size = 6,
  size = 3,
  hjust = 0.75,
  angle = -45,
  palette = "PuOr" # colorblind safe, photocopy-able
) + ggplot2::labs(title = "Points Per Game")
```

---

ggfluctuation2 *Fluctuation plot*

Description

Create a fluctuation plot.

Usage

```
ggfluctuation2(table_data, floor = 0, ceiling = max(table_data$freq, na.rm =
  TRUE))
```

Arguments

table_data	a table of values, or a data frame with three columns, the last column being frequency
floor	don't display cells smaller than this value
ceiling	max value to compare to

Details

A fluctutation diagram is a graphical representation of a contingency table. This fuction currently only supports 2D contingency tables. The function was adopted from experiemntal functions within GGplot2 developed by Hadley Wickham.

Author(s)

Hadley Wickham <h.wickham@gmail.com>, Barret Schloerke <schloerke@gmail.com>

Examples

```
data(movies, package = "ggplot2")
ggfluctuation2(table(movies$Action, movies$Comedy))
ggfluctuation2(table(movies$Action, movies$mpaa))
ggfluctuation2(table(movies[,c("Action", "mpaa")]))
ggfluctuation2(table(warpbreaks$breaks, warpbreaks$tension))
```

---

ggnet *ggnet - Plot a network with ggplot2*

Description

Function for making a network plot from an object of class network or igraph, using ggplot2. Please visit http://github.com/briatte/ggnet for the latest development and descriptions about ggnet.

Usage

```
ggnet(net, mode = "fruchtermanreingold", layout.par = NULL, size = 12,
  alpha = 0.75, weight.method = "none", names = c("", ""),
  node.group = NULL, node.color = NULL, node.alpha = NULL,
  segment.alpha = NULL, segment.color = "grey", segment.label = NULL,
  segment.size = 0.25, arrow.size = 0, label.nodes = FALSE,
  label.size = size/2, top8.nodes = FALSE, trim.labels = TRUE,
  quantize.weights = FALSE, subset.threshold = 0,
  legend.position = "right", ...)
```

Arguments

net	an object of class igraph or network. If the object is of class igraph, the intergraph package is used to convert it to class network.
mode	a placement method from the list of modes provided in the sna package. Defaults to the Fruchterman-Reingold force-directed algorithm. If mode is set to "geo" and net contains two vertex attributes called "lat" and "lon", these are used instead for geographic networks.
layout.par	options to the placement method, as listed in gplot.layout.
size	size of the network nodes. Defaults to 12. If the nodes are weighted, their area is proportionally scaled up to the size set by size.
alpha	a level of transparency for nodes, vertices and arrows. Defaults to 0.75.
weight.method	a weighting method for the nodes. Accepts "indegree", "outdegree" or "degree" (the default). Set to "none" to plot unweighted nodes.

names	a character vector of two elements to use as legend titles for the node groups and node weights. Defaults to empty strings.
node.group	a vector of character strings to label the nodes with, of the same length and order as the vertex names. Factors are converted to strings prior to plotting.
node.color	a vector of character strings to color the nodes with, holding as many colors as there are levels in node.group. Tries to default to "Set1" if missing.
node.alpha	transparency of the nodes. Inherits from alpha.
segment.alpha	transparency of the vertex links. Inherits from alpha.
segment.color	color of the vertex links. Defaults to "grey".
segment.size	size of the vertex links, as a vector of values or as a single value. Defaults to 0.25.
segment.label	labels for the vertex links at mid-edges. Label size will be set to 1 / segment.size, and label alpha will inherit from alpha.
arrow.size	size of the vertex arrows for directed network plotting, in centimeters. Defaults to 0.
label.nodes	label nodes with their vertex names attribute. If set to TRUE, all nodes are labelled. Also accepts a vector of character strings to match with vertex names.
label.size	size of the labels. Defaults to size / 2.
top8.nodes	use the top 8 nodes as node groups, colored with "Set1". The rest of the network will be plotted as the ninth (grey) group. Experimental.
trim.labels	removes '@', 'http://', 'www.' and the ending '/' from vertex names. Cleans up labels for website and Twitter networks. Defaults to TRUE.
quantize.weights	break node weights to quartiles. Fails when quartiles do not uniquely identify nodes.
subset.threshold	delete nodes prior to plotting, based on weight.method < subset.threshold. If weight.method is unspecified, total degree (Freeman's measure) is used. Defaults to 0 (no subsetting).
legend.position	location of the captions for node colors and weights. Accepts all positions supported by ggplot2 themes. Defaults to "right".
...	other arguments supplied to geom_text for the node labels. Arguments pertaining to the title or other items can be achieved through ggplot2 methods.

Details

The weight.method argument produces visually scaled nodes that are proportionally sized to their unweighted degree. To compute weighted centrality or degree measures, see Tore Opsahl's tnet package.

Author(s)

Moritz Marbach <mmarbach@mail.uni-mannheim.de> and Francois Briatte <f.briatte@ed.ac.uk>

See Also

gplot in the sna package

Examples

```
require(network)
# make toy random network
x                  <- 10
ndyads             <- x * (x - 1)
density            <- x / ndyads
nw.mat             <- matrix(0, nrow = x, ncol = x)
dimnames(nw.mat)   <- list(1:x, 1:x)
nw.mat[row(nw.mat) != col(nw.mat)] <- runif(ndyads) < density
nw.mat
rnd <- network::network(nw.mat)
rnd

# random network
pRnd <- ggnet(rnd, label.nodes = TRUE, alpha = 1, color = "white", segment.color = "grey10")
# pRnd

# random groups
category = LETTERS[rbinom(x, 4, .5)]
ggnet(rnd, label.nodes = TRUE, color = "white", segment.color = "grey10", node.group = category)

# city and service firms data from the UCIrvine Network Data Repository
data(cityServiceFirms, package = "GGally")

# plot cities, firms and law firms
type = cityServiceFirms %v% "type"
type = ifelse(grepl("City|Law", type), gsub("I+", "", type), "Firm")
pRnd <- ggnet(cityServiceFirms, mode = "kamadakawai", alpha = .2, node.group = type,
      label.nodes = c("Paris", "Beijing", "Chicago"), color = "darkred")
# pRnd
```

---

ggpairs *ggpairs - A GGplot2 Matrix*

Description

Make a matrix of plots with a given data set

Usage

```
ggpairs(data, columns = 1:ncol(data), title = "", upper = list(),
  lower = list(), diag = list(), params = NULL, ...,
  axisLabels = "internal", legends = FALSE, verbose = FALSE)
```

Arguments

data	data set using. Can have both numerical and categorical data.
columns	which columns are used to make plots. Defaults to all columns.
title	title for the graph
upper	see Details
lower	see Details
diag	see Details
params	vector of parameters to be applied to geoms. Each value must have a corresponding name, such as `c(binwidth = 0.1)`.
...	other parameters being supplied to geom's aes, such as color
axisLabels	either "internal" for labels in the diagonal plots, "none" for no axis labels, or "show" to display axisLabels
legends	boolean to determine the printing of the legend in each plot. Not recommended.
verbose	boolean to determine the printing of "Plot #1, Plot #2...."

Details

upper and lower are lists that may contain the variables 'continuous', 'combo' and 'discrete'. Each element of the list is a string implementing the following options: continuous = exactly one of ('points', 'smooth', 'density', 'cor', 'blank'); combo = exactly one of ('box', 'dot', 'facethist', 'facetdensity', 'denstrip', 'blank'); discrete = exactly one of ('facetbar','ratio', 'blank').

diag is a list that may only contain the variables 'continuous' and 'discrete'. Each element of the diag list is a string implmenting the following options: continuous = exactly one of ('density', 'bar', 'blank'); discrete = exactly one of ('bar', 'blank').

If a list option it will be set to the function default. If 'blank' is ever chosen as an option, then ggpairs will produce a blank plot, as if nothing was printed there.

Value

ggpair object that if called, will print

Author(s)

Barret Schloerke <schloerke@gmail.com>, Jason Crowley <crowley.jason.s@gmail.com>, Di Cook <dicook@iastate.edu>, Heike Hofmann <hofmann@iastate.edu>, Hadley Wickham <h.wickham@gmail.com>

Examples

```
# plotting is reduced to the first couple of examples.
# Feel free to print the ggpair objects created in the examples

data(tips, package = "reshape")
pm <- ggpairs(tips[,1:3])
# pm
pm <- ggpairs(tips)
# pm
```

```
pm <- ggpairs(tips, upper = "blank")
# pm


# Custom Example
pm <- ggpairs(
  tips[,1:4],
  upper = list(continuous = "density", combo = "box"),
  lower = list(continuous = "points", combo = "dot")
)
# pm

# Use sample of the diamonds data
data(diamonds, package="ggplot2")
diamonds.samp <- diamonds[sample(1:dim(diamonds)[1],200),]

# Custom Example
pm <- ggpairs(
 diamonds.samp[,1:3],
 upper = list(continuous = "density", combo = "box"),
 lower = list(continuous = "points", combo = "dot"),
 color = "cut",
 title = "Diamonds"
)
# pm

# Will plot four "Incorrect Plots"
bad_plots <- ggpairs(
  tips[,1:3],
  upper = list(continuous = "wrongType1", combo = "wrongType2"),
  lower = list(continuous = "IDK1", combo = "IDK2", discrete = "mosaic"),
)
# bad_plots

# Labels on the outside, grids won't line up
pm <- ggpairs(tips[,1:3], axisLabels="none")
# pm

# Custom Examples
custom_car <- ggpairs(mtcars[,c("mpg","wt","cyl")], upper = "blank", title = "Custom Example")
# ggplot example taken from example(geom_text)
  plot <- ggplot2::ggplot(mtcars, ggplot2::aes(x=wt, y=mpg, label=rownames(mtcars)))
  plot <- plot +
    ggplot2::geom_text(ggplot2::aes(colour=factor(cyl)), size = 3) +
    ggplot2::scale_colour_discrete(l=40)
custom_car <- putPlot(custom_car, plot, 1, 2)
personal_plot <- ggally_text(
  "ggpairs allows you\nto put in your\nown plot.\nLike that one.\n <---"
)
custom_car <- putPlot(custom_car, personal_plot, 1, 3)
# custom_car
```

ggparcoord *ggparcoord - A ggplot2 Parallel Coordinate Plot*

Description

A function for plotting static parallel coordinate plots, utilizing the ggplot2 graphics package.

Usage

```
ggparcoord(data, columns, groupColumn = NULL, scale = "std",
  scaleSummary = "mean", centerObsID = 1, missing = "exclude",
  order = columns, showPoints = FALSE, alphaLines = 1, boxplot = FALSE,
  shadeBox = NULL, mapping = NULL, title = "")
```

Arguments

data	the dataset to plot
columns	a vector of variables (either names or indices) to be axes in the plot
groupColumn	a single variable to group (color) by
scale	method used to scale the variables (see Details)
scaleSummary	if scale=="center", summary statistic to univariately center each variable by
centerObsID	if scale=="centerObs", row number of case plot should univariately be centered on
missing	method used to handle missing values (see Details)
order	method used to order the axes (see Details)
showPoints	logical operator indicating whether points should be plotted or not
alphaLines	value of alpha scaler for the lines of the parcoord plot or a column name of the data
boxplot	logical operator indicating whether or not boxplots should underlay the distribution of each variable
shadeBox	color of underlaying box which extends from the min to the max for each variable (no box is plotted if shadeBox == NULL)
mapping	aes string to pass to ggplot object
title	character string denoting the title of the plot

Details

scale is a character string that denotes how to scale the variables in the parallel coordinate plot. Options:

- std: univariately, subtract mean and divide by standard deviation
- robust: univariately, subtract median and divide by median absolute deviation

- uniminmax: univariately, scale so the minimum of the variable is zero, and the maximum is one
- globalminmax: no scaling is done; the range of the graphs is defined by the global minimum and the global maximum
- center: use uniminmax to standardize vertical height, then center each variable at a value specified by the scaleSummary param
- centerObs: use uniminmax to standardize vertical height, then center each variable at the value of the observation specified by the centerObsID param

missing is a character string that denotes how to handle missing missing values. Options:

- exclude: remove all cases with missing values
- mean: set missing values to the mean of the variable
- median: set missing values to the median of the variable
- min10: set missing values to 10% below the minimum of the variable
- random: set missing values to value of randomly chosen observation on that variable

order is either a vector of indices or a character string that denotes how to order the axes (variables) of the parallel coordinate plot. Options:

- (default): order by the vector denoted by columns
- (given vector): order by the vector specified
- anyClass: order variables by their separation between any one class and the rest (as opposed to their overall variation between classes). This is accomplished by calculating the F-statistic for each class vs. the rest, for each axis variable. The axis variables are then ordered (decreasing) by their maximum of k F-statistics, where k is the number of classes.
- allClass: order variables by their overall F statistic (decreasing) from an ANOVA with groupColumn as the explanatory variable (note: it is required to specify a groupColumn with this ordering method). Basically, this method orders the variables by their variation between classes (most to least).
- skewness: order variables by their sample skewness (most skewed to least skewed)
- Outlying: order by the scagnostic measure, Outlying, as calculated by the package scagnostics. Other scagnostic measures available to order by are Skewed, Clumpy, Sparse, Striated, Convex, Skinny, Stringy, and Monotonic. Note: To use these methods of ordering, you must have the scagnostics package loaded.

Value

ggplot object that if called, will print

Author(s)

Jason Crowley <crowley.jason.s@gmail.com>, Barret Schloerke <schloerke@gmail.com>, Di Cook <dicook@iastate.edu>, Heike Hofmann <hofmann@iastate.edu>, Hadley Wickham <h.wickham@gmail.com>

Examples

```
# use sample of the diamonds data for illustrative purposes
data(diamonds, package="ggplot2")
diamonds.samp <- diamonds[sample(1:dim(diamonds)[1],100),]

# basic parallel coordinate plot, using default settings
# ggparcoord(data = diamonds.samp,columns = c(1,5:10))

# this time, color by diamond cut
gpd <- ggparcoord(data = diamonds.samp,columns = c(1,5:10),groupColumn = 2)
# gpd

# underlay univariate boxplots, add title, use uniminmax scaling
gpd <- ggparcoord(data = diamonds.samp,columns = c(1,5:10),groupColumn = 2,
  scale = "uniminmax",boxplot = TRUE,title = "Parallel Coord. Plot of Diamonds Data")
# gpd

# utilize ggplot2 aes to switch to thicker lines
gpd <- ggparcoord(data = diamonds.samp,columns = c(1,5:10),groupColumn = 2,
  title="Parallel Coord. Plot of Diamonds Data",mapping = ggplot2::aes(size = 1))
# gpd

# basic parallel coord plot of the msleep data, using 'random' imputation and
# coloring by diet (can also use variable names in the columns and groupColumn
# arguments)
data(msleep, package="ggplot2")
gpd <- ggparcoord(data = msleep, columns = 6:11, groupColumn = "vore", missing =
  "random", scale = "uniminmax")
# gpd

# center each variable by its median, using the default missing value handler,
# 'exclude'
gpd <- ggparcoord(data = msleep, columns = 6:11, groupColumn = "vore", scale =
  "center", scaleSummary = "median")
# gpd

# with the iris data, order the axes by overall class (Species) separation using
# the anyClass option
gpd <- ggparcoord(data = iris, columns = 1:4, groupColumn = 5, order = "anyClass")
# gpd

# add points to the plot, add a title, and use an alpha scalar to make the lines
# transparent
gpd <- ggparcoord(data = iris, columns = 1:4, groupColumn = 5, order = "anyClass",
  showPoints = TRUE, title = "Parallel Coordinate Plot for the Iris Data",
  alphaLines = 0.3)
# gpd

# color according to a column
iris2 <- iris
iris2$alphaLevel <- c("setosa" = 0.2, "versicolor" = 0.3, "virginica" = 0)[iris2$Species]
gpd <- ggparcoord(data = iris2, columns = 1:4, groupColumn = 5, order = "anyClass",
```

```
  showPoints = TRUE, title = "Parallel Coordinate Plot for the Iris Data",
  alphaLines = "alphaLevel")
# gpd
```

---

ggsurv *Plot* survfit *objects using* ggplot2

Description

This function produces Kaplan-Meier plots using ggplot2. As a first argument it needs a survfit object, created by the survival package. Default settings differ for single stratum and multiple strata objects.

Usage

```
ggsurv(s, CI = "def", plot.cens = TRUE, surv.col = "gg.def",
  cens.col = "red", lty.est = 1, lty.ci = 2, cens.shape = 3,
  back.white = FALSE, xlab = "Time", ylab = "Survival", main = "")
```

Arguments

s	an object of class survfit
CI	should a confidence interval be plotted? Defaults to TRUE for single stratum objects and FALSE for multiple strata objects.
plot.cens	mark the censored observations?
surv.col	colour of the survival estimate. Defaults to black for one stratum, and to the default ggplot2 colours for multiple strata. Length of vector with colour names should be either 1 or equal to the number of strata.
cens.col	colour of the points that mark censored observations.
lty.est	linetype of the survival curve(s). Vector length should be either 1 or equal to the number of strata.
lty.ci	linetype of the bounds that mark the 95% CI.
cens.shape	shape of the points that mark censored observations.
back.white	if TRUE the background will not be the default grey of ggplot2 but will be white with borders around the plot.
xlab	the label of the x-axis.
ylab	the label of the y-axis.
main	the plot label.

Value

An object of class ggplot

Author(s)

Edwin Thoen <edwinthoen@gmail.com>

Examples

```
if (require(survival) && require(scales)) {
  data(lung, package = "survival")
  sf.lung <- survival::survfit(Surv(time, status) ~ 1, data = lung)
  ggsurv(sf.lung)

  # Multiple strata examples
  sf.sex <- survival::survfit(Surv(time, status) ~ sex, data = lung)
  pl.sex <- ggsurv(sf.sex)
  pl.sex

  # Adjusting the legend of the ggsurv fit
  pl.sex +
    ggplot2::guides(linetype = FALSE) +
    ggplot2::scale_colour_discrete(
      name   = 'Sex',
      breaks = c(1,2),
      labels = c('Male', 'Female')
    )

  # We can still adjust the plot after fitting
  data(kidney, package = "survival")
  sf.kid <- survival::survfit(Surv(time, status) ~ disease, data = kidney)
  pl.kid <- ggsurv(sf.kid, plot.cens = FALSE)
  pl.kid

  # Zoom in to first 80 days
  pl.kid <- pl.kid + ggplot2::coord_cartesian(xlim = c(0, 80), ylim = c(0.45, 1))
  pl.kid

  # Add the diseases names to the plot and remove legend
  col <- scales::hue_pal(
    h         = c(0, 360) + 15,
    c         = 100,
    l         = 65,
    h.start   = 0,
    direction = 1
  )(4)
  pl.kid +
    ggplot2::annotate(
      "text",
      label  = c('AN', 'GN', 'Other', 'PKD'),
      x      = c(50, 20, 50, 71),
      y      = c(0.47, 0.55, 0.67, 0.8),
      size   = 5,
      colour = col
    ) +
    ggplot2::guides(color = FALSE, linetype = FALSE)
```

```
}
```

`happy` *Data related to happiness from the General Social Survey, 1972-2006.*

Description

This data extract is taken from Hadley Wickham's `productplots` package. The original description follows, with minor edits.

Usage

```
data(happy)
```

Format

A data frame with 51020 rows and 10 variables

Details

The data is a small sample of variables related to happiness from the General Social Survey (GSS). The GSS is a yearly cross-sectional survey of Americans, run from 1972. We combine data for 25 years to yield 51,020 observations, and of the over 5,000 variables, we select nine related to happiness:

- age. age in years: 18–89.
- degree. highest education: lt high school, high school, junior college, bachelor, graduate.
- finrela. relative financial status: far above, above average, average, below average, far below.
- happy. happiness: very happy, pretty happy, not too happy.
- health. health: excellent, good, fair, poor.
- marital. marital status: married, never married, divorced, widowed, separated.
- sex. sex: female, male.
- wtsall. probability weight. 0.43–6.43.

References

Smith, Tom W., Peter V. Marsden, Michael Hout, Jibum Kim. *General Social Surveys, 1972-2006*. [machine-readable data file]. Principal Investigator, Tom W. Smith; Co-Principal Investigators, Peter V. Marsden and Michael Hout, NORC ed. Chicago: National Opinion Research Center, producer, 2005; Storrs, CT: The Roper Center for Public Opinion Research, University of Connecticut, distributor. 1 data file (57,061 logical records) and 1 codebook (3,422 pp).

putPlot *Put Plot*

Description

Function to place your own plot in the layout.

Usage

```
putPlot(plotMatrix, plotObj, rowFromTop, columnFromLeft)
```

Arguments

plotMatrix	ggally object to be altered
plotObj	ggplot object to be placed
rowFromTop	row from the top
columnFromLeft	column from the left

Author(s)

Barret Schloerke <schloerke@gmail.com>

Examples

```
custom_car <- ggpairs(mtcars[,c("mpg","wt","cyl")], upper = "blank", title = "Custom Example")
# ggplot example taken from example(geom_text)
  plot <- ggplot2::ggplot(mtcars, ggplot2::aes(x=wt, y=mpg, label=rownames(mtcars)))
  plot <- plot +
    ggplot2::geom_text(ggplot2::aes(colour=factor(cyl)), size = 3) +
    ggplot2::scale_colour_discrete(l=40)
custom_car <- putPlot(custom_car, plot, 1, 2)
personal_plot <- ggally_text(
  "ggpairs allows you\nto put in your\nown plot.\nLike that one.\n <---"
)
custom_car <- putPlot(custom_car, personal_plot, 1, 3)
# custom_car
```

scagOrder *Find order of variables*

Description

Find order of variables based on a specified scagnostic measure by maximizing the index values of that measure along the path.

Usage

```
scagOrder(scag, vars, measure)
```

Arguments

scag	scagnostics object
vars	character vector of the variables to be ordered
measure	scagnostics measure to order according to

Value

character vector of variable ordered according to the given scagnostic measure

Author(s)

Jason Crowley <crowley.jason.s@gmail.com>

---

singleClassOrder *Order axis variables*

Description

Order axis variables by separation between one class and the rest (most separation to least).

Usage

```
singleClassOrder(classVar, axisVars, specClass = NULL)
```

Arguments

classVar	class variable (vector from original dataset)
axisVars	variables to be plotted as axes (data frame)
specClass	character string matching to level of classVar; instead of looking for separation between any class and the rest, will only look for separation between this class and the rest

Value

character vector of names of axisVars ordered such that the first variable has the most separation between one of the classes and the rest, and the last variable has the least (as measured by F-statistics from an ANOVA)

Author(s)

Jason Crowley <crowley.jason.s@gmail.com>

skewness *Sample skewness*

Description

Calculate the sample skewness of a vector while ignoring missing values.

Usage

```
skewness(x)
```

Arguments

x	numeric vector

Value

sample skewness of x

Author(s)

Jason Crowley <crowley.jason.s@gmail.com>

Index