

# Package ‘GGally’

January 2, 2012

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

**License** GPL (>= 2.0)

**Title** Extension to ggplot2.

**Type** Package

**LazyLoad** yes

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

**Description** Package contains templates for different plots to be combined into a plot matrix, as well as a parallel coordinate plot function.

**Version** 0.3.1

**Date** 2011-06-30

**Depends** ggplot2, stringr

**Suggests** scagnostics

**Collate** ‘find-combo.r’ ‘gg-plots.r’ ‘ggpairs.r’ ‘ggparcoord.R’

**Repository** CRAN

**Date/Publication** 2011-11-03 21:30:41

## R topics documented:

get.VarTypes . . . . .	2
getPlot . . . . .	3
ggally_barDiag . . . . .	3
ggally_blank . . . . .	4
ggally_box . . . . .	4
ggally_cor . . . . .	5
ggally_density . . . . .	6

ggally_densityDiag . . . . .	6
ggally_denstrip . . . . .	7
ggally_diagAxis . . . . .	8
ggally_dot . . . . .	8
ggally_dotAndBox . . . . .	9
ggally_facetbar . . . . .	10
ggally_facetdensity . . . . .	10
ggally_facetdensitystrip . . . . .	11
ggally_facethist . . . . .	12
ggally_points . . . . .	12
ggally_ratio . . . . .	13
ggally_smooth . . . . .	13
ggally_text . . . . .	14
ggfluctuation2 . . . . .	15
ggpairs . . . . .	15
ggparcoord . . . . .	19
putPlot . . . . .	22
scagOrder . . . . .	22
singleClassOrder . . . . .	23
skewness . . . . .	24

<b>Index</b>	<b>25</b>
--------------	-----------

---

get.VarTypes	<i>Get vector of variable types from data frame</i>
--------------	-----------------------------------------------------

---

## 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	<i>getPlot Retrieves the ggplot object at the desired location</i>
---------	--------------------------------------------------------------------

---

**Description**

getPlot 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**

```
plotMatrix2 <- ggpairs(iris[,5:4], upper = list(combo = "denstrip"))
getPlot(plotMatrix2, 1, 2)
```

---

ggally_barDiag	<i>Plots the Bar Plots by Using Diagonal Plots the bar plots by using Diagonal</i>
----------------	------------------------------------------------------------------------------------

---

**Description**

Plots the Bar Plots by Using Diagonal Plots the bar plots by using Diagonal

**Usage**

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

**Arguments**

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

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_barDiag(movies, aes(x = mpaa))
ggally_barDiag(movies, aes_string(x = "mpaa"))
ggally_barDiag(movies, aes_string(x = "rating", binwidth = ".1"))
```

---

ggally_blank	<i>Blank Drawing nothing</i>
--------------	------------------------------

---

**Description**

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

**Usage**

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

**Arguments**

... other arguments ignored

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

---

ggally_box	<i>Plots the Box Plot Make a box plot with a given data set</i>
------------	-----------------------------------------------------------------

---

**Description**

Plots the Box Plot 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**

```
ggally_box(iris, aes(x = Petal.Width, y = Species))
ggally_box(iris, aes_string(x = "Petal.Width", y = "Species"))
ggally_box(iris, aes_string(y = "Petal.Width", x = "Species", color = "Species"), outlier.colour = "red", outlier.size = 10)
```

---

ggally\_cor

*Correlation from the Scatter Plot estimate correlation from the given data*

---

**Description**

Correlation from the Scatter Plot estimate correlation from the given data

**Usage**

```
ggally_cor(data, mapping, corAlignPercent = 0.6, corSize = 3, ...)
```

**Arguments**

data	data set using
mapping	aesthetics being used
corAlignPercent	right align position of numbers. Default is 60 percent across the horizontal
corSize	size of text
...	other arguments being supplied to geom_text

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_cor(iris, aes(x = Sepal.Length, y = Petal.Length))
ggally_cor(iris, aes_string(x = "Sepal.Length", y = "Petal.Length", size = 15, colour = "red"))
ggally_cor(iris, aes_string(x = "Sepal.Length", y = "Petal.Length", color = "Species"), corSize = 15)
```

---

ggally_density	<i>Plots the Scatter Density Plot Make a scatter density plot from a given data</i>
----------------	-------------------------------------------------------------------------------------

---

**Description**

The aesthetic "fill" determines whether or not stat\_density2d (filled) or geom\_density2d (lines) is used.

**Usage**

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

**Arguments**

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

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_density(iris, aes(x = Sepal.Length, y = Petal.Length))
ggally_density(iris, aes_string(x = "Sepal.Length", y = "Petal.Length"))
ggally_density(iris, aes_string(x = "Sepal.Length", y = "Petal.Length", fill = "..level.."))
ggally_density(iris, aes_string(x = "Petal.Length", y = "Petal.Width", fill = "..level..")) + scale_fill_gradient()
```

---

ggally_densityDiag	<i>Plots the Density Plots by Using Diagonal Plots the density plots by using Diagonal</i>
--------------------	--------------------------------------------------------------------------------------------

---

**Description**

Plots the Density Plots by Using Diagonal 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**

```
ggally_densityDiag(iris, aes(x = Petal.Width))
ggally_densityDiag(movies, aes_string(x="rating"))
ggally_densityDiag(movies, aes_string(x="rating", color = "mpaa"))
```

---

ggally\_denstrip      *Plots a tile plot with facets Make Tile Plot as densely as possible*

---

**Description**

Plots a tile plot with facets 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**

```
ggally_denstrip(iris, aes(x = Petal.Width, y = Species, color = Species))
ggally_denstrip(iris, aes_string(x = "Petal.Width", y = "Species"))
ggally_denstrip(iris, aes_string(x = "Species", y = "Petal.Width", binwidth = "0.2")) + scale_fill_gradient(low = '')
```

---

ggally_diagAxis	<i>Internal Axis Labeling Plot for ggpairs ; is used when axisLabels == "internal"</i>
-----------------	----------------------------------------------------------------------------------------

---

**Description**

Internal Axis Labeling Plot for ggpairs ; is used when axisLabels == "internal"

**Usage**

```
ggally_diagAxis(data, mapping, ...)
```

**Arguments**

data	dataset being plotted
mapping	aesthetics being used (x is the variable the plot will be made for)
...	other arguments for geom_text

**Author(s)**

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

**Examples**

```
ggally_diagAxis(iris,aes(x=Petal.Width))
ggally_diagAxis(iris,aes(x=Species))
```

---

ggally_dot	<i>Plots the Box Plot with Dot Add jittering with the box plot</i>
------------	--------------------------------------------------------------------

---

**Description**

Plots the Box Plot with Dot 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**

```
ggally_dot(iris, aes(x = Petal.Width, y = Species))
ggally_dot(iris, aes_string(x = "Petal.Width", y = "Species"))
ggally_dot(iris, aes_string(x = "Species", y = "Petal.Width", color = "Species"))
ggally_dot(iris, aes_string(x = "Species", y = "Petal.Width", color = "Species", shape = "Species")) + scale_shape()
```

---

ggally_dotAndBox	<i>Plots either Box Plot or Dot Plots Place box plots or dot plots on the graph</i>
------------------	-------------------------------------------------------------------------------------

---

**Description**

Plots either Box Plot or Dot Plots 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**

```
ggally_dotAndBox(iris, aes(x = Petal.Width, y = Species, color = Species), boxPlot=TRUE)
ggally_dotAndBox(iris, aes(x = Petal.Width, y = Species, color = Species), 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**

<code>data</code>	data set using
<code>mapping</code>	aesthetics being used
<code>...</code>	other arguments are sent to <code>geom_bar</code>

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_facetbar(tips, aes(x = sex, y = smoker, fill = time))
ggally_facetbar(tips, aes(x = smoker, y = sex, fill = time))
```

---

ggally\_facetdensity      *Plots the density plots by faceting Make density plots by displaying subsets of the data in different panels*

---

**Description**

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

**Usage**

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

**Arguments**

<code>data</code>	data set using
<code>mapping</code>	aesthetics being used
<code>...</code>	other arguments being sent to <code>stat_density</code>

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_facetdensity(iris, aes(x = Petal.Width, y = Species))
ggally_facetdensity(iris, aes_string(x = "Species", y = "Petal.Width", color = "Species"))
```

---

ggally\_facetdensitystrip

*Plots a density plot with facets or a tile plot with facets Make Tile Plot as densely as possible*

---

**Description**

Plots a density plot with facets or a tile plot with facets 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 decide 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	<i>Plots the Histograms by Faceting Make histograms by displaying subsets of the data in different panels</i>
------------------	---------------------------------------------------------------------------------------------------------------

---

**Description**

Plots the Histograms by Faceting 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**

```
ggally_facethist(iris, aes(x = Petal.Width, y = Species))  
ggally_facethist(iris, aes_string(x = "Species", y = "Petal.Width"), binwidth = 0.1)
```

---

ggally_points	<i>Plots the Scatter Plot</i>
---------------	-------------------------------

---

**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**

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

---

ggally\_ratio

*Plots a mosaic plots Plots the mosaic plot by using fluctuation*

---

**Description**

Must send only two discrete columns in the data set.

**Usage**

```
ggally_ratio(data)
```

**Arguments**

data            data set using

**Author(s)**

Barret Schloerke <schloerke@gmail.com>

**Examples**

```
ggally_ratio(movies[,c("mpaa", "Action")])
ggally_ratio(movies[,c("mpaa", "Action")] + coord_equal())
ggally_ratio(movies[,c("Action", "mpaa")] + opts(aspect.ratio = (length(levels(movies[, "mpaa"])) ) / (length(lev
```

---

ggally\_smooth

*Plots the Scatter Plot with Smoothing Add a smoothed condition mean with a given scatter plot*

---

**Description**

Plots the Scatter Plot with Smoothing 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**

```
ggally_smooth(iris, aes(x = Sepal.Length, y = Sepal.Width))
ggally_smooth(iris, aes_string(x = "Sepal.Length", y = "Sepal.Width"))
ggally_smooth(iris, aes_string(x = "Sepal.Length", y = "Petal.Length", color = "Species"))
```

---

ggally\_text

*GGplot Text Plot text for a plot*

---

**Description**

GGplot Text Plot text for a plot

**Usage**

```
ggally_text(label, mapping = 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 = aes_string(size = 15, color = "red"))
```

---

ggfluctuation2	<i>Fluctuation plot Create a fluctuation plot.</i>
----------------	----------------------------------------------------

---

### Description

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

### 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

### Author(s)

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

### Examples

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

---

ggpairs	<i>ggpairs - A GGplot2 Matrix Make a matrix of plots with a given data set</i>
---------	--------------------------------------------------------------------------------

---

### Description

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', 'facetedensity', 'denstrip', 'blank'); discrete = exactly one of ('facetbar', 'ratio', 'blank').

**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 <code>c(binwidth = 0.1)</code> .
...	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**

diag is a list that may only contain the variables 'continuous' and 'discrete'. Each element of the diag list is a string implementing 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

#ggpairs(iris)
#ggpairs(iris, upper = "blank")
```

```

ggpairs(iris[,3:5])

# Custom Example
ggpairs(
  iris[,3:5],
  upper = list(continuous = "density", combo = "box"),
  lower = list(continuous = "points", combo = "dot"),
  diag = list(continuous = "bar", discrete = "bar")
)

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

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

# Will plot four "Incorrect Plots"
bad_plots <- ggpairs(
  iris[,3:4],
  upper = list(continuous = "wrongType1", combo = "wrongType2"),
  lower = list(continuous = "IDK1", combo = "IDK2", discrete = "mosaic"),
  diag = list(continuous = "points", discrete = "box")
)
#bad_plots

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

# Custom plot with different scale fill gradient
custom_fill <- ggpairs(iris[,5:4], upper = list(combo = "denstrip"))
custom_fill <- putPlot(
  custom_fill,
  ggally_text("ggpairs allows you\nto retrieve a plot.\nWe will grab this one,\n-->\nwith the legend\nand axis labels
1, 1)
#custom_fill

```

```

plot <- getPlot(custom_fill, 1, 2)
#plot
plotNew <- plot + scale_fill_gradient(low = "grey80", high = "black")
#plotNew
custom_fill <- putPlot(custom_fill, plotNew, 1, 2)
#custom_fill

special_diamond <- ggpairs(
  diamonds.samp,
  columns = 8:10,
  upper = list(continuous = "points", aes_string = aes_string(color = "clarity")),
  lower = list(continuous = "points", aes_string = aes_string(color = "cut")),
  diag = "blank",
  title = "Diamonds"
)
#special_diamond

## prints
# data = mtcars
# columns = c(1,3,4) # (mpg, disp, hp)
# upper = contains scatter plots with the shape defined by the cyl and size constant at 5
# lower = contains scatter plots with the size defined by the cyl
# diag = contains 'blank' plots
# color = defined by cyl and is applied to both the upper and lower plots. It would be applied to diag if it existed
# title = "Custom Cars"
# verbose = FALSE makes the "Plot #1, Plot #2..." not print
carsMatrix <- ggpairs(
  mtcars,
  columns = c(1,3,4),
  upper = list(continuous = "points", aes_string = aes_string(shape = "cyl", size = 5)),
  lower = list(continuous = "points", aes_string = aes_string(size = "cyl")),
  diag = "blank",
  color = "cyl",
  title = "Custom Cars",
  verbose = FALSE
)
#carsMatrix

## Each list is custom to its own area
# params are the parameters applied to the geoms. Parameters can not be considered aesthetics.
iris_with_params <- ggpairs(
  iris,
  upper = list(continuous = "density", combo = "dot", aes_string = aes_string(color = "Species")),
  lower = list(continuous = "smooth", combo = "denstrip", aes_string = aes_string(color = "Species", fill = "Species")),
  diag = list(continuous = "bar", discrete = "bar", aes_string = aes_string(fill = "Species"), params = c(binwidth = 0.1)),
  title = "Complex Iris Data"
)
#iris_with_params

## The same plot matrix with no axis labels

```

```
iris_with_params2 <- ggpairs(
  iris,
  upper = list(continuous = "density", combo = "dot", aes_string = aes_string(color = "Species")),
  lower = list(continuous = "smooth", combo = "denstrip", aes_string = aes_string(color = "Species", fill = "Species")),
  diag = list(continuous = "bar", discrete = "bar", aes_string = aes_string(fill = "Species"), params = c(binwidth = 0.1)),
  title = "Complex Iris Data",
  axisLabels = "none"
)
#iris_with_params2
```

---

ggparcoord

*ggparcoord - A ggplot2 Parallel Coordinate Plot ; A function for plotting static parallel coordinate plots, utilizing the ggplot2 graphics package.*

---

### Description

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

### 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
boxplot	logical operator indicating whether or not boxplots should underlay the distribution of each variable
shadeBox	color of underlying 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

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
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
ggparcoord(data = diamonds.samp,columns = c(1,5:10),groupColumn = 2)

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

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

# 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)
ggparcoord(data = msleep, columns = 6:11, groupColumn = "vore", missing =
"random", scale = "uniminmax")

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

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

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

---

putPlot	<i>Put Plot Function to place your own plot in the layout</i>
---------	---------------------------------------------------------------

---

**Description**

Put Plot 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**

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

---

scagOrder	<i>Find order of variables based on a specified scagnostic measure by maximizing the index values of that measure along the path.</i>
-----------	---------------------------------------------------------------------------------------------------------------------------------------

---

**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	<i>Order axis variables by separation between one class and the rest (most separation to least)</i>
------------------	-----------------------------------------------------------------------------------------------------

---

**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	<i>Calculate the sample skewness of a vector while ignoring missing values.</i>
----------	---------------------------------------------------------------------------------

---

**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

## \*Topic **hplot**

- getPlot, 3
- ggally\_barDiag, 3
- ggally\_blank, 4
- ggally\_box, 4
- ggally\_cor, 5
- ggally\_density, 6
- ggally\_densityDiag, 6
- ggally\_denstrip, 7
- ggally\_dot, 8
- ggally\_dotAndBox, 9
- ggally\_facetbar, 10
- ggally\_facetdensity, 10
- ggally\_facetdensitystrip, 11
- ggally\_facethist, 12
- ggally\_points, 12
- ggally\_ratio, 13
- ggally\_smooth, 13
- ggally\_text, 14
- ggfluctuation2, 15
- ggpairs, 15
- putPlot, 22

  

- ggally\_ratio, 13
- ggally\_smooth, 13
- ggally\_text, 14
- ggfluctuation2, 15
- ggpairs, 15
- ggparcoord, 19

  

- putPlot, 22

  

- scagOrder, 22
- singleClassOrder, 23
- skewness, 24

  

- get.VarTypes, 2
- getPlot, 3
- ggally\_barDiag, 3
- ggally\_blank, 4
- ggally\_box, 4
- ggally\_cor, 5
- ggally\_density, 6
- ggally\_densityDiag, 6
- ggally\_denstrip, 7
- ggally\_diagAxis, 8
- ggally\_dot, 8
- ggally\_dotAndBox, 9
- ggally\_facetbar, 10
- ggally\_facetdensity, 10
- ggally\_facetdensitystrip, 11
- ggally\_facethist, 12
- ggally\_points, 12