# Package: MVQuickGraphs (via r-universe)

September 8, 2024

**Type** Package

**Title** Quick Multivariate Graphs

**Version** 0.1.6

**Date** 2021-06-23

**Description** Functions used for graphing in multivariate contexts.
These functions are designed to support produce reasonable
graphs with minimal input of graphing parameters. The
motivation for these functions was to support students learning
multivariate concepts and R - there may be other functions and
packages better-suited to practical data analysis. For details
about the ellipse methods see Johnson and Wichern (2007,
ISBN:9780131877153).

**License** GPL-2 | GPL-3

**Encoding** UTF-8

**Imports** stats, graphics, plotrix, corrplot

**Suggests** psych

**RoxygenNote** 7.1.1

**Repository** https://douglaswhitaker.r-universe.dev

**RemoteUrl** https://github.com/douglaswhitaker/mvquickgraphs

**RemoteRef** HEAD

**RemoteSha** 68f612f445f2a6abfcd10c2d860db7e0c938c127

## Contents

1

---

bvNormalContour                    *Bivariate Normal Contour Ellipse*

---

### Description

Draws a contour of constant density at the (1-alpha)100% level for a bivariate normal distribution
using the eigendecomposition of the covariance matrix. This is likely more interesting for learning
about the bivariate normal distribution than as a practical tool, for which other functions already
exist (e.g. link[graphics]{contour}).

### Usage

```
bvNormalContour(
  mu = c(0, 0),
  Sigma = NULL,
  eig = NULL,
  xl = NULL,
  yl = NULL,
  axes = TRUE,
  center = FALSE,
  lim.adj = 0.02,
  alpha = 0.05,
  ...
)
```

### Arguments

| | |
|---|---|
| mu | a vector giving the mean of the bivariate normal distribution. This is the center of the ellipse. |
| Sigma | a matrix giving the covariance matrix of the bivariate normal distribution. Either Sigma or eig must be specified. |
| eig | the eigenvalues and eigenvectors of the covariance matrix. This should be of the same form as the output of [eigen](#), namely a list with two components: values and vectors. It is assumed that the largest eigenvalue is given first. Either Sigma or eig must be specified. |
| xl | a vector giving the lower and upper limits of the x-axis for plotting. If xl = NULL (default), then reasonable values are computed automatically. |
| yl | a vector giving the lower and upper limits of the y-axis for plotting. If yl = NULL (default), then reasonable values are computed automatically. |
| axes | logical. If axes = TRUE (default) then the major and minor axes of the ellipse are plotted. |
| center | logical. If axes = TRUE then the center of the ellipse is indicated with a point and dashed lines are drawn to the x-axis and y-axis. |

| | |
|---|---|
| lim.adj | a value giving an adjustment to the x-axis and y-axis limits computed if either `xl = NULL` or `yl = NULL`. Essentially this is a way to have some coarse control over these limits for quick graphing: positive values will increase the distance between the upper and lower limits (making the ellipse appear smaller) while negative values will decrease the distance (and make the ellipse appear larger). |
| alpha | a value giving the value of alpha to be used when computing the contour. Contours are drawn at the `1-alpha` level. |
| ... | other arguments to be passed to the graphing functions. |

## Value

None

## References

Johnson, R. A., & Wichern, D. W. (2007). Applied multivariate statistical analysis (6th ed). Pearson Prentice Hall.

## Examples

```
mu <- c(-1,8)
Sigma <- matrix(c(3,2,2,4), ncol = 2)
# Draw a 90% contour
bvNormalContour(mu = mu, Sigma = Sigma, alpha = 0.10)
```

---

| | |
|---|---|
| confidenceEllipse | *Bivariate Normal Confidence Ellipse* |

---

## Description

Draws a (1-alpha)100% confidence ellipse (two dimensional) for a multivariate normal distribution using the eigendecomposition of the covariance matrix.

## Usage

```
confidenceEllipse(
  X.mean = c(0, 0),
  eig,
  n,
  p,
  xl = NULL,
  yl = NULL,
  axes = TRUE,
  center = FALSE,
  lim.adj = 0.02,
  alpha = 0.05,
  ...
)
```

## Arguments

| | |
|---|---|
| `X.mean` | a column matrix giving the mean of the two dimensions of the p-dimensional multivariate normal distribution. |
| `eig` | the eigenvalues and eigenvectors of the covariance matrix. This should be of the same form as the output of [`eigen`](), namely a list with two components: `values` and `vectors`. It is assumed that the largest eigenvalue is given first. |
| `n` | the number of observations. |
| `p` | the number of dimensions of the multivariate normal distribution. (The resulting graph will always be a two-dimensional confidence region for the two dimensions of a p-dimensional multivaraite normal distribution under consideration.) |
| `xl` | a vector giving the lower and upper limits of the x-axis for plotting. If `xl = NULL` (default), then reasonable values are computed automatically. |
| `yl` | a vector giving the lower and upper limits of the y-axis for plotting. If `yl = NULL` (default), then reasonable values are computed automatically. |
| `axes` | logical. If `axes = TRUE` (default) then the major and minor axes of the ellipse are plotted. |
| `center` | logical. If `axes = TRUE` then the center of the ellipse is indicated with a point and dashed lines are drawn to the x-axis and y-axis. |
| `lim.adj` | a value giving an adjustment to the x-axis and y-axis limits computed if either `xl = NULL` or `yl = NULL`. Essentially this is a way to have some coarse control over these limits for quick graphing: positive values will increase the distance between the upper and lower limits (making the ellipse appear smaller) while negative values will decrease the distance (and make the ellipse appear larger). |
| `alpha` | a value giving the value of alpha to be used when computing the contour. Contours are drawn at the `1-alpha` level. |
| `...` | other arguments to be passed to the graphing functions. |

## Value

None

## References

Johnson, R. A., & Wichern, D. W. (2007). Applied multivariate statistical analysis (6th ed). Pearson Prentice Hall.

## Examples

```
# 90% Confidence Ellipse for Reading and Vocab from ability.cov
x.bar <- ability.cov$center[5:6]
Sigma <- ability.cov$cov[5:6,5:6]
n <- ability.cov$n.obs
p <- length(ability.cov$center)

confidenceEllipse(X.mean = x.bar,
                  eig = eigen(Sigma),
```

```
                n = n, p = p,
                alpha = 0.10)
```

---

eigenEllipseHelper      *Helper Function for other Ellipse-from-Eigendecomposition Func-*
                        *tions*

---

### Description

Helper function for graphing ellipses from eigendecompositions. This function is used by bvNormalContour and confidenceEllipse. Essentially this is a wrapper for draw.ellipse that also calculates appropriate x-axis and y-axis limits to make graphing an ellipse easier (because the entire ellipse should be visible without any work on the user's part to specify the limits).

### Usage

```
eigenEllipseHelper(mu, lengths, angle, xl, yl, lim.adj, axes, center, ...)
```

### Arguments

| | |
|---|---|
| mu | column matrix giving the coordinates for the cener of the ellipse. |
| lengths | vector giving the major and minor axis lengths. |
| angle | angle of rotation (in radians). |
| xl | x-axis limits. If xl = NULL then these are computed automatically. |
| yl | y-axis limits. If yl = NULL then these are computed automatically. |
| lim.adj | a value giving an adjustment to the x-axis and y-axis limits computed if either xl = NULL or yl = NULL. |
| axes | logical. If axes = TRUE, then the major and minor axes are graphed. |
| center | logical. If axes = TRUE then the center of the ellipse is indicated with a point and dashed lines are drawn to the x-axis and y-axis. |
| ... | other arguments to be passed to the graphing functions. |

### Value

None

---

| make_all_rects | *Allows the creation of all rectangles on a correlation plot using a vector of cutpoints to divide them.* |

---

### Description

Allows the creation of all rectangles on a correlation plot using a vector of cutpoints to divide them.

### Usage

```
make_all_rects(
  cutpoints,
  endpoint,
  ondiag = TRUE,
  offdiag = TRUE,
  col_ondiag = "black",
  col_offdiag = "red",
  ondiag_width = 4,
  offdiag_width = 3,
  correlation_matrix,
  ...
)
```

### Arguments

| | |
|---|---|
| cutpoints | A vector containing the names of the cutpoints (i.e names of variables contained in the correlation plot). The first cutpoint is the variable that will start off the first rectangle. The second cutpoint then defines the starting point of the second rectangle, and so on. |
| endpoint | The name of the variable contained in the correlation plot that will define the end of the last rectangle to be made. |
| ondiag | Defaults to TRUE. If TRUE, the function will generate the associated on-diagonal rectangles. |
| offdiag | Defaults to TRUE. If FALSE, the function will generate the associated off-diagonal rectangles. |
| col_ondiag | Defines the colour of on-diagonal rectangles. |
| col_offdiag | Defines the colour of off-diagonal rectangles. |
| ondiag_width | Defines the line width of the on-diagonal rectangles. |
| offdiag_width | Defines the line width of the off-diagonal rectangles. |
| correlation_matrix | |
| | The correlation matrix the correlation plot is based on. |
| ... | Arguments to modify graphical parameters, etc. |

## Examples

```
#Adds all rectangles associated with the cutpoints to the correlation plot of
#the Bechtoldt sample correlation matrix (provided by the psych package).
library(psych)
corrplot::corrplot(Bechtoldt)
make_all_rects(cutpoints=c("First_Names", "Vocabulary", "Suffixes"), endpoint
="Three_Higher", correlation_matrix=Bechtoldt)

#Adds all on-diagonal rectangles associated with the cutpoints.
make_all_rects(cutpoints=c("First_Names", "Vocabulary", "Suffixes"), endpoint
="Three_Higher", offdiag=FALSE, correlation_matrix=Bechtoldt)
```

---

make_rect                          *Add a rectangle to a correlation plot*

---

## Description

Add a rectangle to a correlation plot

## Usage

```
make_rect(
  rstart,
  rend,
  cstart = NULL,
  cend = NULL,
  correlation_matrix,
  mirror = FALSE,
  lwd = 3,
  ...
)
```

## Arguments

| | |
|---|---|
| rstart | The name of the variable contained in the correlation plot that will serve as the vertical starting point of the rectangle. |
| rend | The name of the variable contained in the correlation plot that will serve as the vertical ending point of the rectangle. |
| cstart | (optional) The name of the variable contained in the correlation plot that will serve as the horizontal starting point of the rectangle. If no cstart or cend provided, creates an on-diagonal rectangle automatically. cstart and cend allows for off-diagonal rectangles. |
| cend | (optional) The name of the variable contained in the correlation plot that will serve as the horizontal ending point of the rectangle. If no cstart or cend provided, creates an on-diagonal rectangle automatically. cstart and cend allows for off-diagonal rectangles. |

correlation_matrix

> The correlation matrix the correlation plot is based on.

mirror              If TRUE, also adds the equivalent rectangle from the other side of the diagonal (i.e. the "mirror" of the original). The function ignores this entirely if the rectangle is on the diagonal, as there is no mirror.

lwd                  Determines the width of the rectangle lines.

...                   Arguments to modify graphical parameters, etc.

### Examples

```
#Adding an on-diagonal rectangle to a correlation plot with mirroring, using
#the Bechtoldt sample correlation matrix provided by the psych package.
library(corrplot)
library(psych)
corrplot(Bechtoldt)
make_rect(rstart="First_Names", rend="Flags", correlation_matrix=Bechtoldt, mirror=TRUE)

#Adding an off-diagonal rectangle
make_rect(rstart="First_Names", rend="Flags", cstart="First_Names",
cend="Sentences", correlation_matrix=Bechtoldt)
```

---

   plot4in1                       *Plot 4-in-1*

---

### Description

Generates a 2x2 panel graph including four residual diagnostic plots as is popular in some other statistics packages. This was initially written to support students learning R for the first time in a regression modeling course. `plot4in1` generates four commonly-used residual diagnostic plots that can be used to assess the linear regression assumptions and ensures a consistent, reasonably-pleasing graphical style across each plot.

### Usage

```
plot4in1(
  out,
  type = "Regular",
  PP = TRUE,
  pch = 19,
  col = "steelblue",
  cex = 1.2,
  ...
)
```

## Arguments

| | |
|---|---|
| out | the output of the [lm](#) function (an object of class `"lm"`). The components of greatest importance from this object are residuals (perhaps passed to rstandard of rstudent, depending on `type`) and `fitted.values`. |
| type | the type of residuals to be used. There are three possible values: `"Regular"`, `"Standardized"`, and `"Studentized"`. Using `type = "Regular"` results in untransformed residuals being used, `type = "Standardized"` uses standardized residuals (computed using rstandard), and `type = "Studentized"` uses externally studentized residuals (computed using rstudent). |
| PP | logical. If PP = TRUE, a Normal Percentile Plot (P-P Plot) is displayed in the top-left panel. If PP = FALSE, a Normal Quantile Plot (Q-Q Plot) is displayed in the top-left panel. |
| pch | symbol to be used in plotting. pch = 19 is a filled circle (see [par](#)). |
| col | color of symbol specified in pch to be used in graphing. The default is `"steelblue"` (see [par](#)). |
| cex | character expansion value, used to adjust the size of the symbol specified in pch. The default value is cex = 1.2 (see [par](#)). |
| ... | other arguments to be passed to the graphing functions. |

## Details

plot4in1 creates a 2 by 2 panel using par(mfrow = c(2,2)) and then generates four residual diagnostic plots: a Percentile-Percentile (or Quantile-Quantile plot if PP = FALSE), a scatterplot of the `fitted.values` against the residuals, a histogram of the residuals, and scatterplot of the residuals against their order, overplotted.

## Value

None

## See Also

[influence.measures](#) for more information about standardized (rstandard) and studentized (rstudent) residuals; [qqnorm](#) for more information about the Quantile-Quanitle (Q-Q) plot; [par](#) for information about the graphical parameters.

## Examples

```
out <- lm(Girth ~ Volume, data = trees)
plot4in1(out)
```

# Index