Package ‘prettyGraphs’

July 23, 2025

Type Package

Title Publication-Quality Graphics

Version 2.2.0

Date 2025-03-30

Description Simple and crisp publication-quality graphics for the ExPosition family of packages. See An ExPosition of the Singular Value Decomposition in R (Beaton et al 2014) <doi:10.1016/j.csda.2013.11.006>.

License GPL-2

Encoding UTF-8

BugReports `https://github.com/derekbeaton/ExPosition1/issues`

RoxygenNote 7.3.2

NeedsCompilation no

Author Derek Beaton [aut, cre]

Maintainer Derek Beaton <`exposition.software@gmail.com`>

Repository CRAN

Date/Publication 2025-04-12 18:50:01 UTC

Contents

add.alpha 2
contributionBars 2
correlationPlotter 3
createColorVectorsByDesign 5
minmaxHelper 5
peeledHull 6
prettyBars 7
prettyGraphs 9
prettyGraphsColors 10
prettyGraphsColorSelection 10
prettyGraphsHSVColorSelection 11
prettyPlot 12
prettyScree 14
repmat 16

Index 17

---

add.alpha *add.alpha*

Description

A function to add alpha levels to RGB colors.

Usage

```
add.alpha(col, alpha = 0.65)
```

Arguments

`col`	color from colors()
`alpha`	alpha level (between 0 and 1)

Note

This code was created by Markus Gesmann. Derek Beaton included the code in prettyGraphs because it is a versatile function and used in prettyGraphs. See: https://magesblog.com/post/2013-04-30-how-to-change-alpha-value-of-colours-in/

Author(s)

Markus Gesmann

---

contributionBars *contributionBars*

Description

Produces bar charts for multivariate analyses. Plots the contribution to the variance from each data point for upwards of two axes (components).

Usage

```
contributionBars(
  factor_scores,
  contributions,
  x_axis = 1,
  y_axis = 2,
  col = NULL,
  main = NULL,
  upper = "steelblue4",
  lower = "firebrick2",
```

```
    threshold = 0,
    sortContributions = TRUE,
    pretty = FALSE,
    show.bg.bars = FALSE
  )
```

Arguments

`factor_scores`	The factor scores, or x- and y-axis points of a data set.
`contributions`	The amount of contribution to the variance (between 0-1) by each data point.
`x_axis`	Which axis is the x-axis? Default is 1.
`y_axis`	Which axis is the y-axis? Default is 2.
`col`	A single-column matrix of colors for each data point.
`main`	A title to be placed at the top of the graph.
`upper`	The color used to identify the upper bound items that contribute above average variance.
`lower`	The color used to identify the lower bound items that contribute above average variance.
`threshold`	A threshold (between 0-1) to draw upper and lower bounds for important items. Default is 1/number of items.
`sortContributions`	boolean, if TRUE, items will be sorted by contribution. if FALSE, items will appear in their row order.
`pretty`	a boolean. If FALSE, use the current version. If TRUE, make the bars with `prettyBars`.
`show.bg.bars`	a boolean. Only used if `pretty` is TRUE. If TRUE, background bars are plotted for a fill effect.

Author(s)

Derek Beaton

`correlationPlotter` *correlationPlotter*

Description

Computes and plots a correlation circle (used in multivariate analyses). Correlation is computed between measured items and components (factors, dimensions, principal axes, etc...).

Usage

```
correlationPlotter(
  data_matrix,
  factor_scores,
  x_axis = 1,
  y_axis = 2,
  col = NULL,
  pch = NULL,
  xlab = "",
  ylab = "",
  main = "",
  axis.lwd = 3,
  circle.lwd = 3,
  circle.col = "#00000040",
  asp = 1,
  dev.new = TRUE
)
```

Arguments

`data_matrix`	A set of data (i.e., original measures and observations)
`factor_scores`	One set of factor scores that were computed from the original data matrix.
`x_axis`	Which axis is the x-axis? Default is 1.
`y_axis`	Which axis is the y-axis? Default is 2.
`col`	A single-column matrix of colors for each data point.
`pch`	A single-column matrix of pch for each data point. Indicates which point style to use for each item. See `par`.
`xlab`	A label to be placed along the x-axis.
`ylab`	A label to be placed along the y-axis.
`main`	A title to be placed at the top of the graph.
`axis.lwd`	numeric. Line width for the axes.
`circle.lwd`	numeric. Line width for the circle.
`circle.col`	color for the circle
`asp`	numeric. Aspect ratio (see `asp` in `par`).
`dev.new`	boolean. If TRUE, `dev.new` is called internally to create new device. If FALSE, a device must already be open.

Author(s)

Derek Beaton

`createColorVectorsByDesign` *createColorVectorsByDesign*

Description

Produces a color vector for items by using a design matrix.

Usage

```
createColorVectorsByDesign(design_matrix, hsv = TRUE, offset = NULL)
```

Arguments

`design_matrix`	A dummy coded design matrix.
`hsv`	a boolean. If TRUE, use `prettyGraphsHSVColorSelection`. If FALSE, use `prettyGraphsColorSelection`.
`offset`	numeric. An offset value to be passed to `prettyGraphsHSVColorSelection` or `prettyGraphsColorSelection`.

Value

Produces a list with the following items

`oc`	The colors of the observations (based on group)
`gc`	The colors of the groups

Author(s)

Derek Beaton

`minmaxHelper` *minmaxHelper*

Description

Computes the x- and y- axis constraints for all prettyGraphs functions.

Usage

```
minmaxHelper(mat1, mat2 = NULL, axis1 = 1, axis2 = 2, findBounds = TRUE)
```

Arguments

`mat1`	Matrix that will be plotted. Must be at least 2 rows by 2 columns.
`mat2`	if NULL, mat2=mat2. Second matrix that will be plotted. Must be at least 2 rows by 2 columns.
`axis1`	Which column of the matrix will be the x-axis? Default = 1.
`axis2`	Which column of the matrix will be the y-axis? Default = 2.
`findBounds`	a boolean. If TRUE, min and max values are found for positive and negative on `axis1` and `axis2`. If FALSE, one value is used for all constraints.

Value

`minMaxList`	A list with the following values: minx, miny, maxx, maxy

Author(s)

Derek Beaton

Examples

```
data(beavers)
minMaxList <- minmaxHelper(as.matrix(beaver1),as.matrix(beaver1),1,2)
```

`peeledHull` *peeledHull*

Description

Computes and plots a peeled hull around a set of points onto a current graphics device.

Usage

```
peeledHull(
  data_matrix,
  x_axis = 1,
  y_axis = 2,
  percentage = 1,
  col = "black",
  lwd = 3,
  lty = 1
)
```

Arguments

`data_matrix`	A set of data you would like to plot on 2 dimensions (e.g., a scatter plot). Can be original data or factor scores or anything with at least 2 columns.
`x_axis`	Which axis is the x-axis? Default is 1.
`y_axis`	Which axis is the y-axis? Default is 2.
`percentage`	The percentage of points that should be enveloped by the hull.
`col`	The color of the hull (see col in plot()).
`lwd`	The thickness of the hull line (see lwd in plot())
`lty`	The line type (see lty in plot()).

Note

The code for this function was (barely) adapted from Michael Greenacre's book on correspondence analysis. All credit for this code should go to Michael Greenacre; I only turned it into a function. The original code can be found at http://carme-n.org/?sec=code2

Author(s)

Derek Beaton turned Michael Greenacre's code into a function.

References

Greenacre, M. J. (2007). Correspondence Analysis in Practice. *Chapman and Hall*.

`prettyBars` *prettyBars*

Description

prettyBars creates simple, crisp, publication-style quality bar graphs.

Usage

```
prettyBars(
  data,
  axis = 1,
  cex.names = 0.5,
  fg.col = NULL,
  axis.lims = NULL,
  show.bg.bars = FALSE,
  threshold.line = FALSE,
  main = "",
  bg.border = "white",
  bg.col = NULL,
  bg.lims = NULL,
```

```
  sort.data = TRUE,
  horiz = TRUE,
  dev.new = TRUE
)
```

Arguments

`data`	A set of data you would like to plot with bars for 1 dimension. Can be original data or factor scores or anything with at least 1 column.
`axis`	which axis (column of `data`) should be plotted?
`cex.names`	size of the text
`fg.col`	a matrix (single column) of colors for bars corresponding to rows of `data`.
`axis.lims`	two values in the form of `c(min,max)` for plot limits. If NULL, the min and max are computed.
`show.bg.bars`	a set of bars to be plotted under the bars in `data`. Used for a fill effect.
`threshold.line`	boolean. If TRUE, a line perpendicular to the bars is plotted. The lines appear at `bg.lims`.
`main`	A title to be placed at the top of the graph.
`bg.border`	color for `show.bg.bars` borders (see `border` in `barplot`)
`bg.col`	a matrix (single column) of colors for background bars to be plotted under `fg.col`.
`bg.lims`	two values in the form of `c(min,max)` for where to plot background bars or threshold line.
`sort.data`	a boolean. Will sort the bars by descending values.
`horiz`	see `horiz` in `barplot`.
`dev.new`	boolean. If TRUE, `dev.new` is called internally to create new device. If FALSE, a device must already be open.

Value

`bp.cols`	locations of bars (as would be returned by `barplot`).

Author(s)

Derek Beaton

See Also

`barplot`, `prettyPlot`

Examples

```
##stolen from ?barplot
#tN <- table(Ni <- stats::rpois(100, lambda = 5))
#the.colors <- rainbow(length(tN))
#dev.new()
#r <- barplot(tN, col = the.colors)
#prettyBars(as.matrix(tN),axis=1,fg.col=as.matrix(the.colors),horiz=FALSE,dev.new=TRUE)
```

`prettyGraphs` *prettyGraphs: for publication-quality graphics.*

Description

prettyGraphs is a package that produces simple, crisp, publication-quality graphics for multivariate analyses.

Author(s)

Derek Beaton <exposition.software@gmail.com>

References

Three functions were copied/derived for use in prettyGraphs: peeledHull, add.alpha, and repmat.

For peeledHull see:
http://carme-n.org/?sec=code2
Greenacre, M. J. (2007). Correspondence Analysis in Practice. *Chapman and Hall.*

For repmat see:
http://cran.r-project.org/doc/contrib/R-and-octave.txt

For add.alpha see:
https://magesblog.com/post/2013-04-30-how-to-change-alpha-value-of-colours-in/

See Also

`prettyPlot` `contributionBars` `correlationPlotter` `peeledHull` `minmaxHelper` `repmat`

`prettyGraphsColors` *prettyGraphsColors*

Description

A function to return only a subset of R colors used by the prettyGraphs package.

Usage

```
prettyGraphsColors()
```

Value

`possibleColors`	A list of colors used by prettyGraphs

Author(s)

Derek Beaton

`prettyGraphsColorSelection`

Algorithmically select colors from prettyGraphs

Description

This function uses prime numbers to select colors from `prettyGraphsColors`.

Usage

```
prettyGraphsColorSelection(n.colors = 1, offset = NULL, starting.color = 163)
```

Arguments

`n.colors`	number of colors needed
`offset`	numeric. Should be a prime number, if it is not, the closest prime is selected. This number decides the distance between indices for color selection.
`starting.color`	numeric. This is the starting location (e.g., color) in a matrix of `prettyGraphsColors`.

Value

a `matrix`	a matrix of colors are returned.

Author(s)

Derek Beaton

prime number selection from here:
http://stackoverflow.com/questions/3789968/generate-a-list-of-primes-in-r-up-to-a-certain-number

See Also

`prettyGraphsColors`, `prettyGraphsHSVColorSelection`

`prettyGraphsHSVColorSelection`

Algorithmically select HSV colors.

Description

This function uses an HSV color selection algorithm to create HSV color palletes.

Usage

```
prettyGraphsHSVColorSelection(
  n.colors = 1,
  offset = NULL,
  h = 13,
  s = 0.75,
  v = 0.75
)
```

Arguments

`n.colors`	number of colors needed
`offset`	numeric (decimal/percentage). This number decides the distance between indices for color selection. If NULL, the golden ratio is selected.
`h`	numeric. The initial hue (see `hsv`).
`s`	numeric. The initial saturation (see `hsv`).
`v`	numeric. The initial value (see `hsv`).

Value

a `matrix`	a matrix of colors are returned.

Author(s)

Derek Beaton

HSV selection from here:
http://martin.ankerl.com/2009/12/09/how-to-create-random-colors-programmatically/

See Also

`prettyGraphsColors`, `prettyGraphsColorSelection`

`prettyPlot` *prettyPlot*

Description

prettyPlot creates simple, crisp, publication-style quality graphics for multivariate analyses.

Usage

```
prettyPlot(
  data_matrix,
  x_axis = 1,
  y_axis = 2,
  col = NULL,
  pch = NULL,
  cex = NULL,
  text.cex = NULL,
  pos = 3,
  xlab = "",
  ylab = "",
  main = "",
  display_names = TRUE,
  display_points = TRUE,
  constraints = NULL,
  contributionCircles = FALSE,
  contributions = NULL,
  axes = TRUE,
  fg.line.width = 3,
  fg.type = "l",
  fg.col = "black",
  bg.line.width = 1.5,
  bg.lty = 3,
  bg.col = "black",
  flip = FALSE,
  asp = 1,
  findBounds = TRUE,
  dev.new = TRUE,
  new.plot = TRUE
)
```

Arguments

`data_matrix`	A set of data you would like to plot on 2 dimensions (e.g., a scatter plot). Can be original data or factor scores or anything with at least 2 columns.

`x_axis`	Which axis is the x-axis? Default is 1.
`y_axis`	Which axis is the y-axis? Default is 2.
`col`	any acceptable color format. A single-column matrix of colors for each data point. A single value can be used.
`pch`	A single-column matrix of pch for each data point. A single value can be used. Indicates which point style to use for each item. See `par`.
`cex`	a single column of continuous values. A single value can be used. Indicates the size of the points. See `par`. Used multiplicatively for points with `contributions`.
`text.cex`	A replacement for the cex parameter in text() (to avoid collisions with cex). Used additively for points with `contributions`. See `par` and `text`.
`pos`	integer between 1-4. Determines position of text for points. See `par`.
`xlab`	A label to be placed along the x-axis.
`ylab`	A label to be placed along the y-axis.
`main`	A title to be placed at the top of the graph.
`display_names`	boolean. If TRUE, the names of the points will be displayed on the plot.
`display_points`	boolean. If TRUE, the points will be displayed on the plot.
`constraints`	A list that contains the plot constraints. Default is NULL (constraints are computed each time).
`contributionCircles`	boolean. If TRUE, items plotted will vary in size, dependent on amount of contribution to the variance.
`contributions`	A matrix of contribution values for data_matrix (should also be the same size & dimensionality as data_matrix)
`axes`	boolean. If TRUE, a new set of axes are plotted.
`fg.line.width`	integer. Determines thickness of foreground (default: solid) axis lines, see `points` and `lwd` under `par`.
`fg.type`	character or string Determines type of points for foreground (default: solid) axis lines, see `points` and `type` under `par`.
`fg.col`	any acceptable color format. Determines color for foreground (default: solid) axis lines, see `points` and `col` under `par`.
`bg.line.width`	integer. Determines thickness of background (default: dashed) axis lines, see `abline` and `lwd` under `par`.
`bg.lty`	integer. Determines type of background (default: dashed) axis lines, see `abline` and `lty` under `par`.
`bg.col`	any acceptable color format. Determines color of background (default: dashed) axis lines, see `abline` and `col` under `par`.
`flip`	boolean. If TRUE, exchanges bg and col for all applicable pch values.
`asp`	numeric. Aspect ratio, see `par`.
`findBounds`	boolean. If TRUE, finds the min and max of each plotted axes. If FALSE, the largest value is used for all constraints.
`dev.new`	boolean. If TRUE, `dev.new` is called internally to create new device. If FALSE, a device must already be open.
`new.plot`	boolean. If TRUE, `plot` is called for a new plot on the current device. If FALSE, items will be overlayed much like `points`.

Details

All items after contributions (i.e., axes onward) are parameters for very fine-grained detail. They are quite powerful but in most cases not required.

Value

Returns a three item list:

`col`	A matrix of colors where each element is the color for each item plotted.
`pch`	A matrix of pch values where each element is the pch number for each item plotted.
`constraints`	A list (from `minmaxHelper`) of the plot constraints (i.e., min and max for axes)

Author(s)

Derek Beaton

`prettyScree` *prettyScree*

Description

prettyScree creates simple, crisp, publication-style scree plots and “tests” for SVD-based analyses.

Usage

```
prettyScree(
  eigs,
  retain.col = "mediumorchid4",
  dismiss.col = "gray",
  perc.exp = 1,
  n.comps = NULL,
  broken.stick = TRUE,
  kaiser = TRUE,
  main = ""
)
```

Arguments

`eigs`	a vector of *positive* eigenvalues.
`retain.col`	a color for components that are kept.
`dismiss.col`	a color for components that are dismissed.
`perc.exp`	a value between 0 and 1. Used to retain components that explain `perc.comp` * 100 variance. Note: perc.exp retains `cumsum(explained variance) < (perc.exp * 100)` + 1 component.

`n.comps`	a value between 1 and `length(eigs)`. Used to retain `n.comps` number of components.
`broken.stick`	a boolean. If TRUE (default), the broken-stick test is performed (see details).
`kaiser`	a boolean. If TRUE (default), all components with eigenvalues greater than the `mean(eigs)` are retained.
`main`	A title to be placed at the top of the graph.

Details

prettyScree visualizes the distribution of eigenvalues/explained variance from SVD-based analyses. Further, prettyScree performs several rudimentary "tests". Two rudimentary criteria are implemented: 1) user selected explained variance, and 2) user selected number of components. Additionally, two rudimentary "tests" are implemented: 1) the "broken-stick" distribution of variance model, and 2) the "Kaiser criterion" where all components that explain more variance than the mean are kept.

prettyScree colors components that "pass" all selected tests with `retain.col`. Any components that do not pass tests are colored by a more transparent version of `retain.col`. Components that do not meet any criteria for retention are colored by `dismiss.col`.

prettyScree should be considered "under development" as of 12.09.2013. The function works, but we will be adding more features in the near future.

Value

`comps.tests`	a matrix with boolean values. Rows indicate which criteria are used, and columns correspond to components. If TRUE, a component is considered "retained" by that test/criteria. If FALSE, the component is "dismissed" according to that test/criteria.

Note

A private function (`add.alpha`) was copied from http://lamages.blogspot.com/2013/04/how-to-change-alpha-value-of-colours-in.html

Author(s)

Derek Beaton

References

Cangelosi, R., & Goriely, A. (2007). Component retention in principal component analysis with application to cDNA microarray data. *Biology direct*, *2*(2), 1–21.

Peres-Neto, P. R., Jackson, D. A., & Somers, K. M. (2005). How many principal components? Stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, *49*(4), 974–997.

See Also

Also see (some of) the other packages that perform a wide array of SVD-based analyses: `ExPosition`, `FactoMineR`, `ca`, `ade4`

Examples

```
prcomp.res <- prcomp(USArrests, scale = TRUE)
prettyScree(prcomp.res$sdev^2)
##or
princomp.res <- princomp(USArrests, cor = TRUE)
prettyScree(princomp.res$sdev^2)
```

`repmat` *repmat*

Description

A function to match repmat in Matlab (Mathworks, Natick, MA)

Usage

```
repmat(a, n, m)
```

Arguments

`a`	Matrix or vector to repeat.
`n`	Number of row repititions.
`m`	Number of column repititions.

Note

This code was created by Robin Hankin. Derek Beaton included the code in prettyGraphs because it is a versatile function and used in prettyGraphs.

Author(s)

Robin Hankin.

References

For repmat see:
http://cran.r-project.org/doc/contrib/R-and-octave.txt

Index

∗ **bootstrap**
  prettyBars, 7
∗ **graphs**
  contributionBars, 2
  correlationPlotter, 3
  peeledHull, 6
  prettyBars, 7
  prettyGraphs, 9
  prettyGraphsColorSelection, 10
  prettyGraphsHSVColorSelection, 11
  prettyPlot, 12
∗ **misc**
  add.alpha, 2
  createColorVectorsByDesign, 5
  minmaxHelper, 5
  prettyGraphsColors, 10
  repmat, 16
∗ **multivariate**
  contributionBars, 2
  correlationPlotter, 3
  peeledHull, 6
  prettyBars, 7
  prettyGraphs, 9
  prettyPlot, 12
  prettyScree, 14
∗ **package**
  prettyGraphs, 9

abline, *13*
add.alpha, 2

barplot, *8*

contributionBars, 2, *9*
correlationPlotter, 3, *9*
createColorVectorsByDesign, 5

dev.new, *4*, *8*, *13*

hsv, *11*

minmaxHelper, 5, *9*, *14*

par, *4*, *13*
peeledHull, 6, *9*
plot, *13*
points, *13*
prettyBars, *3*, 7
prettyGraphs, 9
prettyGraphs-package (prettyGraphs), 9
prettyGraphsColors, *10*, 10, *11*, *12*
prettyGraphsColorSelection, *5*, 10, *12*
prettyGraphsHSVColorSelection, *5*, *11*, 11
prettyPlot, *8*, *9*, 12
prettyScree, 14

repmat, *9*, 16

text, *13*