Package 'openintro'

January 2, 2012

Type Package

Title OpenIntro data sets and supplement functions

Version 1.2

Date 2011-04-25

Author David M Diez, Christopher D Barr, and Mine Cetinkaya

Maintainer David M Diez <david.m.diez@gmail.com>

Description This package is a supplement to the OpenIntro open source book project. The package contains data sets used in the book along with plotting functions to reproduce the book's figures. Note that many functions and examples include color transparency, which may result in partially some plotting elements not showing up properly in Windows (see Details section of the openintro package help file for details).

License GPL-2

LazyLoad yes

URL http://www.openintro.org/

Repository CRAN

Date/Publication 2011-04-26 16:09:19

R topics documented:

`openintro-package` *OpenIntro book project supplement*

Description

This package is a supplement to the OpenIntro open source book project (http://www.openintro.org/). The package contains data sets used in the book along with plotting functions to reproduce the book's figures. Note that many functions and examples include color transparency, which may result in partially some plotting elements not showing up properly in Windows (see Details section of the openintro package help file for details).

Details

Package:	openintro
Type:	Package
Version:	1.2
Date:	2011-04-25
License:	GPL-2
LazyLoad:	yes

boxPlot, buildAxis, densityPlot, dotPlot, edaPlot, histPlot, normTail, cars, marioKart, possum, run10, satGPA, textbooks

Some colors include transparency, which means they will not be plotted in some operating systems (e.g. Windows). However, the plots may be viewed if they are written to a PDF or PNG file first. For a discussion of this topic, please see

http://yihui.name/en/2007/09/semi-transparent-colors-in-r-color-image-as-an-example/

Two new functions, myPDF and myPNG, were created in this package and may also be used to set up nice plotting files that allow for transparency.

Author(s)

David M Diez and Christopher D Barr

Maintainer: DM Diez <david.m.diez@gmail.com>

Examples

```
#===> boxPlot <===#
data(run10)
par(mfrow=1:2)
boxPlot(run10$time)
boxplot(run10$time)

#===> histPlot, example 1 <===#
data(run10)
par(mfrow=c(2,2))
histPlot(run10$time)
histPlot(run10$time[run10$gender=='M'], probability=TRUE, xlim=c(30, 180),
ylim=c(0, 0.025), hollow=TRUE)
histPlot(run10$time[run10$gender=='F'], probability=TRUE, add=TRUE,
hollow=TRUE, lty=3, border='red')
legend('topleft', col=c('black', 'red'), lty=2:3, legend=c('M','F'))
histPlot(run10$time, col=fadeColor('yellow', '33'), border='darkblue',
probability=TRUE, breaks=30, lwd=3)
brks <- c(40, 50, 60, 65, 70, 75, 80, seq(82.5, 120, 2.5), 125,
130, 135, 140, 150, 160, 170)
histPlot(run10$time, probability=TRUE, breaks=brks,
col=fadeColor('darkgoldenrod4', '33'))
```

```
#===> histPlot, example 2 <===#
data(cars)
par(mfrow=c(1,1))
histPlot(cars$price[cars$type=='small'], probability=TRUE, hollow=TRUE,
xlim=c(0,50))
histPlot(cars$price[cars$type=='midsize'], probability=TRUE, hollow=TRUE,
add=TRUE, border='red', lty=3)
histPlot(cars$price[cars$type=='large'], probability=TRUE, hollow=TRUE,
add=TRUE, border='blue', lty=4)
legend('topright', lty=2:4, col=c('black', 'red', 'blue'),
legend=c('small', 'midsize', 'large'))

#===> densityPlot <===#
data(tips)
par(mfrow=c(1,1))
densityPlot(tips$tip, tips$day)
legend('topright', col=c('black','red'), lty=1:2,
legend=c('Tuesday', 'Friday'))

#===> identifying reasons for outliers <===#
data(marioKart)
par(mfrow=c(1,1))
boxPlot(marioKart$totalPr, marioKart$cond, horiz=TRUE)
these <- which(marioKart$totalPr > 80)
# see the data collection criteria for
# why these observations do not belong.
lines(rep(marioKart$totalPr[these[1]], 2), c(2.4, 2))
text(marioKart$totalPr[these[1]], 2.4, marioKart$title[these[1]],
pos=3, cex=0.5)
lines(rep(marioKart$totalPr[these[2]], 2), c(1.6, 2))
text(marioKart$totalPr[these[2]], 1.6, marioKart$title[these[2]],
pos=1, cex=0.5)

#===> compare plotting methods <===#
data(cars)
par(mfrow=c(1,1))
histPlot(cars$price, ylim=c(0, 0.1), axes=FALSE, ylab='',
probability=TRUE, xlab='price')
axis(1)
boxPlot(cars$price, width=0.03, horiz=TRUE, add=0.067, axes=FALSE)
dotPlot(cars$price, at=0.095, add=TRUE)
densityPlot(cars$price, add=TRUE)

#===> controlling the number of axis labels <===#
#     specify the number of labels
data(textbooks)
x <- textbooks$diff
par(mfrow=c(3,1))
histPlot(x, axes=FALSE)
buildAxis(1, x, n=4, nMin=4, nMax=4)
histPlot(x, axes=FALSE)
buildAxis(1, x, n=5, nMin=5, nMax=5)
histPlot(x, axes=FALSE)
```

```
# no decent axis is found for this data with exactly six labels
# no min or max specified, only a target number of labels:
buildAxis(1, x, n=6)

#===> creating normal plots with tails <===#
par(mfrow=c(2,3), mar=c(3,3,1,1), mgp=c(1.7, 0.7, 0))
normTail(L=-2)
normTail(U=1, xLab='symbol', cex.axis=0.7)
normTail(M=c(-2,-0.3), col='#22558833')
normTail(5, 13, L=-5, M=c(0,3), U=12, xAxisIncr=2)
normTail(102, 4, xlim=c(97,110), M=c(100,103))
normTail(-10.0, 5.192, M=c(-5,2), digits=1, xAxisIncr=2)

#===> Exploratory Data Analysis Plot <===#
data(mlbBat10)
#edaPlot(mlbBat10)
```

`ballBearing` *Lifespan of ball bearings*

Description

A simulated data set on lifespan of ball bearings.

Usage

```
data(ballBearing)
```

Format

A data frame with 75 observations on the following variable.

`lifeSpan` Lifespan of ball bearings (in hours).

Source

Simulated data.

Examples

```
data(ballBearing)
par(mfrow=c(1,2))
histPlot(ballBearing$lifeSpan, col='#22558833')
qqnorm(ballBearing$lifeSpan)
```

boxPlot *Box plot*

Description

An alternative to boxplot. Equations are not accepted. Instead, the second argument, fact, is used to split the data.

Usage

```
boxPlot(x, fact = NULL, horiz = FALSE, width = 2/3, lwd = 1,
lcol = "black", medianLwd = 2, pch = 20, pchCex = 1.8,
col = fadeColor("black", "44"), add=FALSE, key=NULL,
axes = TRUE, xlab = "", ylab = "", xlim = NULL,
ylim = NULL, ...)
```

Arguments

x	A numerical vector.
fact	A character or factor vector defining the grouping for side-by-side box plots.
horiz	If TRUE, the box plot is oriented horizontally.
width	The width of the boxes in the plot. Value between 0 and 1.
lwd	Width of lines used in box and whiskers.
lcol	Color of the box, median, and whiskers.
medianLwd	Width of the line marking the median.
pch	Plotting character of outliers.
pchCex	Size of outlier character.
col	Color of outliers.
add	If FALSE, a new plot is created. Otherwise, the boxplots are added to the current plot for values of TRUE or a numerical vector specifying the locations of the boxes.
key	The order in which to display the side-by-side boxplots. If locations are specified in add, then the elements of add will correspond to the elements of key.
axes	Whether to plot the axes.
xlab	Label for the x axis.
ylab	Label for the y axis.
xlim	Limits for the x axis.
ylim	Limits for the y axis.
...	Additional arguments to plot.

Author(s)

David Diez

See Also

histPlot, dotPlot, densityPlot

Examples

```
data(run10)
par(mfrow=1:2)

#===> comparison 1 <===#
boxPlot(run10$time)
boxplot(run10$time)

#===> comparison 2 <===#
boxPlot(run10$time, run10$gender, col=fadeColor('black', '22'))
boxplot(run10$time ~ run10$gender)

#===> modifications using boxPlot <===#
par(mfrow=c(2,2))
boxPlot(run10$time, run10$gender)
boxPlot(run10$time, run10$gender, xlab='gender',
ylab='run time (min)',
col=fadeColor('black', '18'))
plot(0,0, xlab='gender', ylab='run time (min)',
xlim=c(0,6), ylim=c(30, 180), axes=FALSE)
boxPlot(run10$time, run10$gender, width=0.5, lwd=2,
lcol=4, medianLwd=4, pch=1, pchCex=1, col=4,
add=c(1,2,5), key=c('M','F','N'))
plot(0,0, ylab='gender', xlab='run time (min)',
xlim=c(30, 180), ylim=c(0, 3), axes=FALSE)
boxPlot(run10$time, run10$gender, horiz=TRUE,
xlab='run time (min)', ylab='gender',
add=1:2, key=c('F','M'))
# 'key' can be used to restrict to only the
# desired groups

#===> combine boxPlot and dotPlot <===#
data(tips)
par(mfrow=c(1,1))
boxPlot(tips$tip, tips$day, horiz=TRUE, key=c('Tuesday', 'Friday'))
dotPlot(tips$tip, tips$day, add=TRUE, at=1:2+0.05,
key=c('Tuesday', 'Friday'))

#===> adding a box <===#
par(mfrow=1:2)
boxPlot(run10$time[run10$gender=='M'], xlim=c(0,3))
boxPlot(run10$time[run10$gender=='F'], add=2, axes=FALSE)
axis(1, at=1:2, labels=c('M', 'F'))
boxPlot(run10$time[run10$gender=='M'], ylim=c(0,3), horiz=TRUE)
boxPlot(run10$time[run10$gender=='F'], add=2, horiz=TRUE, axes=FALSE)
axis(2, at=1:2, labels=c('M', 'F'))
```

`buildAxis` *Axis function substitute*

Description

The function `buildAxis` is built to provide more control of the number of labels on the axis. This function is still under development.

Usage

```
buildAxis(side, limits, n, nMin = 2, nMax = 10, extend = 2,
eps = 10^-12, ...)
```

Arguments

`side`	The side of the plot where to add the axis.
`limits`	Either lower and upper limits on the axis or a data set.
`n`	The preferred number of axis labels.
`nMin`	The minimum number of axis labels.
`nMax`	The maximum number of axis labels.
`extend`	How far the axis may extend beyond `range(limits)`.
`eps`	The smallest increment allowed.
`...`	Arguments passed to `axis`

Details

The primary reason behind building this function was to allow a plot to be created with similar features but with different data sets. For instance, if a set of code was written for one data set and the function `axis` had been utilized with pre-specified values, the axis may not match the plot of a new set of data. The function `buildAxis` addresses this problem by allowing the number of axis labels to be specified and controlled.

The axis is built by assigning penalties to a variety of potential axis setups, ranking them based on these penalties and then selecting the axis with the best score.

Value

A vector of the axis plotted.

Author(s)

David M Diez

See Also

`histPlot`, `dotPlot`, `boxPlot`, `densityPlot`

Examples

```
#===> 0 <===#
limits <- rnorm(100, 605490, 10)
hist(limits, axes=FALSE)
buildAxis(1, limits, 2, nMax=4)

#===> 1 <===#
x <- seq(0, 500, 10)
y <- 8*x+rnorm(length(x), mean=6000, sd=200)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=5)
buildAxis(2, limits=y, n=3)

#===> 2 <===#
x <- 9528412 + seq(0, 200, 10)
y <- 8*x+rnorm(length(x), mean=6000, sd=200)
plot(x, y, axes=FALSE)
temp <- buildAxis(1, limits=x, n=4)
buildAxis(2, y, 3)

#===> 3 <===#
x <- seq(367, 1251, 10)
y <- 7.5*x+rnorm(length(x), mean=6000, sd=800)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=4, nMin=3, nMax=3)
buildAxis(2, limits=y, n=4, nMin=3, nMax=5)

#===> 4 <===#
x <- seq(367, 367.1, 0.001)
y <- 7.5*x+rnorm(length(x), mean=6000, sd=0.01)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=4, nMin=5, nMax=6)
buildAxis(2, limits=y, n=2, nMin=3, nMax=4)

#===> 5 <===#
x <- seq(-0.05, -0.003, 0.0001)
y <- 50 + 20*x + rnorm(length(x), sd=0.1)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=4, nMin=5, nMax=6)
buildAxis(2, limits=y, n=4, nMax=5)
abline(lm(y ~ x))

#===> 6 <===#
x <- seq(-0.0097, -0.008, 0.0001)
y <- 50 + 20*x + rnorm(length(x), sd=0.1)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=4, nMin=2, nMax=5)
buildAxis(2, limits=y, n=4, nMax=5)
abline(lm(y ~ x))

#===> 7 <===#
x <- seq(0.03, -0.003099, -0.00001)
```

```
y <- 50 + 20*x + rnorm(length(x), sd=0.1)
plot(x, y, axes=FALSE)
buildAxis(1, limits=x, n=4, nMin=2, nMax=5)
buildAxis(2, limits=y, n=4, nMax=6)
abline(lm(y ~ x))

#===> 8 - repeat <===#
m <- runif(1)/runif(1) +
rgamma(1, runif(1)/runif(1), runif(1)/runif(1))
s <- rgamma(1, runif(1)/runif(1), runif(1)/runif(1))
x <- rnorm(50, m, s)
hist(x, axes=FALSE)
buildAxis(1, limits=x, n=5, nMin=4, nMax=6, eps=10^-12)
if(diff(range(x)) < 10^-12){
cat("too small\n")
}
```

`cars` *cars*

Description

A data frame with 54 rows and 6 columns. The columns represent the variables `type`, `price`, `mpgCity`, `driveTrain`, `passengers`, `weight` for a sample of 54 cars from 1993. This data is a subset of the `Cars93` data set from the `MASS` package.

Usage

```
data(cars)
```

Format

A data frame with 54 observations on the following 6 variables.

`type` The vehicle type with levels `large`, `midsize`, and `small`.

`price` Vehicle price (USD).

`mpgCity` Vehicle mileage in city (miles per gallon).

`driveTrain` Vehicle drive train with levels `4WD`, `front`, and `rear`.

`passengers` The vehicle passenger capacity.

`weight` Vehicle weight (lbs).

Details

These cars represent a random sample for 1993 models that were in both *Consumer Reports* and *PACE Buying Guide*. Only vehicles of type `'small'`, `'midsize'`, and `'large'` were include.

Further description can be found in Lock (1993). Use the URL http://www.amstat.org/publications/jse/v1n1/datasets.lock.html.

Source

Lock, R. H. (1993) 1993 New Car Data. *Journal of Statistics Education* 1(1).

References

http://www.openintro.org/

Examples

```
data(cars)

#===> vehicle price by type <===#
par(mfrow=c(1,1))
histPlot(cars$price[cars$type=='small'], probability=TRUE,
hollow=TRUE, xlim=c(0,50))
histPlot(cars$price[cars$type=='midsize'], probability=TRUE,
hollow=TRUE, add=TRUE, border='red', lty=3)
histPlot(cars$price[cars$type=='large'], probability=TRUE,
hollow=TRUE, add=TRUE, border='blue', lty=4)
legend('topright', lty=2:4, col=c('black', 'red', 'blue'),
legend=c('small', 'midsize', 'large'))

#===> vehicle price versus weight <===#
plot(cars$weight, cars$price, col=fadeColor('magenta', '88'),
pch=20, cex=2)

#===> mileage versus weight <===#
plot(cars$weight, cars$mpgCity, type="n")
temp <- c(seq(1000, 5000, 100), rev(seq(1000, 5000, 100)), 1000)
hold <- 87.11 - 0.03508*temp + 0.000004432*temp^2 + 7*c(rep(-1, 41), rep(1, 41), 1)
polygon(temp, hold, col="#E2E2E2",
border=fadeColor('black', '00'))
points(cars$weight, cars$mpgCity,
col='chocolate4', pch=20, cex=2)
```

---

census *Random sample of 2000 U.S. Census Data*

Description

A random sample of 500 observations from the 2000 U.S. Census Data.

Usage

```
data(census)
```

Format

A data frame with 500 observations on the following 8 variables.

`censusYear` Census Year.

`stateFIPScode` Name of state.

`totalFamilyIncome` Total family income (in U.S. dollars).

`age` Age.

`sex` Sex with levels `Female` and `Male`.

`raceGeneral` Race with levels `American Indian or Alaska Native`, `Black`, `Chinese`, `Japanese`, `Other Asian or Pacific Islander`, `Two major races`, `White` and `Other`.

`maritalStatus` Marital status with levels `Divorced`, `Married/spouse absent`, `Married/spouse present`, `Never married/single`, `Separated` and `Widowed`.

`totalPersonalIncome` Total personal income (in U.S. dollars).

Source

http://factfinder.census.gov/

Examples

```
data(census)
str(census)
these <- census[,3] > 0     # income greater than 0
histPlot(log(census$totalFamilyIncome[these]), xlab="log(total family income)")
```

---

credits *College credits.*

Description

A simulated data set of number of credits taken by college students each semester.

Usage

```
data(credits)
```

Format

A data frame with 100 observations on the following variable.

`credits` Number of credits.

Source

Simulated data.

Examples

```
data(credits)
histPlot(credits$credits)
```

`densityPlot` *Density plot*

Description

Compute kernel density plots, written in the same structure as `boxPlot`. Histograms can be automatically added for teaching purposes.

Usage

```
densityPlot(x, fact = NULL, bw = "nrd0",
histo = c("none", "faded", "hollow"),
breaks = "Sturges", fading = "0E", fadingBorder = "25",
lty = NULL, lwd = 1, col = c("black", "red", "blue"),
key = NULL, add = FALSE, adjust = 1,
kernel = c("gaussian", "epanechnikov", "rectangular",
"triangular", "biweight", "cosine", "optcosine"),
weights = NULL, n = 512, from, to, na.rm = FALSE,
xlim = NULL, ylim = NULL, main = "", ...)
```

Arguments

`x`	A numerical vector.
`fact`	A character or factor vector defining the grouping for data in `x`.
`bw`	Bandwidth. See `density`.
`histo`	Whether to plot a faded histogram (`'faded'`) or hollow histogram (`'hollow'`) in the background. By default, no histogram will be plotted.
`breaks`	The `breaks` argument for histPlot if `histo` is `'faded'` or `'hollow'`.
`fading`	Character value of hexadecimal, e.g. `'22'` or `'5D'`, describing the amount of fading inside the rectangles of the histogram if `histo='faded'`.
`fadingBorder`	Character value of hexadecimal, e.g. `'22'` or `'5D'`, describing the amount of fading of the rectangle borders of the histogram if `histo` is `'faded'` or `'hollow'`.
`lty`	Numerical vector describing the line type for the density curve(s). Each element corresponds to a different level of the argument`fact`.
`lwd`	Numerical vector describing the line width for the density curve(s). Each element corresponds to a different level of the argument`fact`.
`col`	Numerical vector describing the line color for the density curve(s). Each element corresponds to a different level of the argument`fact`.
`key`	An argument to specify ordering of the factor levels.

add	If TRUE, the density curve is added to the plot.
adjust	Argument passed to density to adjust the bandwidth.
kernel	Argument passed to density to select the kernel used.
weights	Argument passed to density to weight observations.
n	Argument passed to density to specify the detail in the density estimate.
from	Argument passed to density specifying the lowest value to include in the density estimate.
to	Argument passed to density specifying the largest value to include in the density estimate.
na.rm	Argument passed to density specifying handling of NA values.
xlim	x-axis limits.
ylim	y-axis limits.
main	Title for the plot.
...	If add=FALSE, then additional arguments to plot.

Author(s)

David Diez

See Also

histPlot, dotPlot, boxPlot

Examples

```
data(tips)
par(mfrow=c(2,2))
histPlot(tips$tip[tips$day == 'Tuesday'], hollow=TRUE, xlim=c(0, 30),
lty=1, main='Tips by day')
histPlot(tips$tip[tips$day == 'Friday'], hollow=TRUE, border='red',
add=TRUE, main='Tips by day')
legend('topright', col=c('black', 'red'), lty=1:2,
legend=c('Tuesday', 'Friday'))
densityPlot(tips$tip, tips$day, col=c('black', 'red'), main='Tips by day')
legend('topright', col=c('black', 'red'), lty=1:2,
legend=c('Tuesday', 'Friday'))
data(run10)
densityPlot(run10$time, histo='faded', breaks=15, main='Run time')
densityPlot(run10$time, histo='hollow', breaks=30, fadingBorder='66',
lty=1, main='Run time')
```

dotPlot *Dot plot*

Description

Plot observations as dots.

Usage

```
dotPlot(x, fact = NULL, vertical = FALSE, at = 1, key = NULL,
pch = 20, col = fadeColor("black", "66"), cex = 1.5,
add = FALSE, axes = TRUE, xlim = NULL, ylim = NULL, ...)
```

Arguments

x	A numerical vector.
fact	A character or factor vector defining the grouping for data in x.
vertical	If TRUE, the plot will be oriented vertically.
at	The vertical coordinate of the points, or the horizontal coordinate if vertical=TRUE. If fact is provided, then locations can be specified for each group.
key	The factor levels corresponding to at, pch, col, and cex.
pch	Plotting character. If fact is given, then different plotting characters can be specified for each factor level. If key is specified, the elements of pch will correspond to the elements of key.
col	Plotting character color. If fact is given, then different colors can be specified for each factor level. If key is specified, the elements of col will correspond to the elements of key.
cex	Plotting character size. If fact is given, then different character sizes can be specified for each factor level. If key is specified, the elements of cex will correspond to the elements of key.
add	If TRUE, then the points are added to the plot.
axes	If FALSE, no axes are plotted.
xlim	Limits for the x axis.
ylim	Limits for the y axis.
...	Additional arguments to be passed to plot if add=FALSE or points if add=TRUE.

Author(s)

David Diez

See Also

histPlot, densityPlot, boxPlot

Examples

```
#===> example 1 <===#
data(cars)
dotPlot(cars$price, cars$type, key=c('large', 'midsize', 'small'), cex=1:3)

#===> example 2 <===#
data(run10)
layout(matrix(1:2,2), heights=c(2.7,1.5))
par(las=1)
these <- run10$gender=='M'
dotPlot(run10$time[these], run10$div[these],
col=fadeColor('black', '11'))
# disorganized levels in the above plot, which we could
# organize with key. an example of organizing the levels...
dotPlot(run10$time[these], run10$div[these],
col=fadeColor('black', '11'),
key=c('20-24', '25-29', '30-34', '35-39'))
par(las=0, mfrow=c(1,1))

#===> example 3 <===#
data(marioKart)
dotPlot(marioKart$totalPr, marioKart$cond, ylim=c(0.5,2.5),
xlim=c(25, 80), cex=1) # miss the outliers
boxPlot(marioKart$totalPr, marioKart$cond, add=1:2+0.1,
key=c('new', 'used'), horiz=TRUE, axes=FALSE)
```

`edaPlot` *Exploratory data analysis plot*

Description

Explore different plotting methods using a click interface.

Usage

```
edaPlot(dataFrame, Col=c('#888888', '#FF0000', '#222222',
                         '#FFFFFF', '#CCCCCC', '#3377AA'))
```

Arguments

`dataFrame`	A data frame.
`Col`	A vector containing six colors. The colors may be given in any form.

Author(s)

David Diez

See Also

histPlot, densityPlot, boxPlot, dotPlot

Examples

```
data(mlbBat10)
bat <- mlbBat10[mlbBat10$AB > 200,]
#edaPlot(bat)

data(marioKart)
mk <- marioKart[marioKart$totalPr < 100,]
#edaPlot(mk)
```

---

fadeColor *Fade colors*

Description

Fade colors so they are transparent.

Usage

```
fadeColor(col, fade = "FF")
```

Arguments

col	An integer, color name, or RGB hexadecimal.
fade	The amount to fade col. This value should be a character in hexadecimal from '00' to 'FF'. The smaller the value, the greater the fading.

Author(s)

David Diez

References

http://research.stowers-institute.org/efg/R/Color/Chart/

See Also

dotPlot

Examples

```
data(marioKart)
new  <- marioKart$cond == 'new'
used <- marioKart$cond == 'used'

par(mfrow=1:2)

#===> color numbers <===#
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80), pch=20,
col=2, cex=2, main='using regular colors')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE, col=4, pch=20, cex=2)
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80),
col=fadeColor(2, '22'), pch=20, cex=2,
main='fading the colors first')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE,
col=fadeColor(4, '22'), pch=20, cex=2)

#===> color names <===#
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80), pch=20,
col='red', cex=2, main='using regular colors')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE, col='blue', pch=20, cex=2)
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80),
col=fadeColor('red', '22'), pch=20, cex=2,
main='fading the colors first')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE,
col=fadeColor('blue', '22'), pch=20, cex=2)

#===> hexadecimal <===#
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80), pch=20,
col='#FF0000', cex=2, main='using regular colors')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE, col='#0000FF', pch=20,
cex=2)
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80),
col=fadeColor('#FF0000', '22'), pch=20, cex=2,
main='fading the colors first')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE,
col=fadeColor('#0000FF', '22'), pch=20, cex=2)

#===> alternative: rgb function <===#
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80), pch=20,
col=rgb(1,0,0), cex=2, main='using regular colors')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE, col=rgb(0,0,1),
pch=20, cex=2)
dotPlot(marioKart$totalPr[new], ylim=c(0,3), xlim=c(25, 80),
col=rgb(1,0,0,1/8), pch=20, cex=2,
main='fading the colors first')
dotPlot(marioKart$totalPr[used], at=2, add=TRUE,
col=rgb(0,0,1,1/8), pch=20, cex=2)
```

---

govRace10 *Election results for 2010 Governor races in the U.S.*

Description

Election results for 2010 Governor races in the U.S.

Usage

```
data(govRace10)
```

Format

A data frame with 37 observations on the following 23 variables.

`id` Unique identifier for the race, which does not overlap with other 2010 races (see `houseRace10` and `senateRace10`)

`state` State name

`abbr` State name abbreviation

`name1` Name of the winning candidate

`perc1` Percentage of vote for winning candidate (if more than one candidate)

`party1` Party of winning candidate

`votes1` Number of votes for winning candidate

`name2` Name of candidate with second most votes

`perc2` Percentage of vote for candidate who came in second

`party2` Party of candidate with second most votes

`votes2` Number of votes for candidate who came in second

`name3` Name of candidate with third most votes

`perc3` Percentage of vote for candidate who came in third

`party3` Party of candidate with third most votes

`votes3` Number of votes for candidate who came in third

`name4` Name of candidate with fourth most votes

`perc4` Percentage of vote for candidate who came in fourth

`party4` Party of candidate with fourth most votes

`votes4` Number of votes for candidate who came in fourth

`name5` Name of candidate with fifth most votes

`perc5` Percentage of vote for candidate who came in fifth

`party5` Party of candidate with fifth most votes

`votes5` Number of votes for candidate who came in fifth

Source

Data was collected from MSNBC.com on November 9th, 2010.

Examples

```
data(govRace10)
table(govRace10[,c("party1", "party2")])
```

gradesTV *Simulated data for analyzing the relationship between watching TV and grades*

Description

This is a simulated data set to be used to estimate the relationship between number of hours per week students watch TV and the grade they got in a statistics class.

Usage

```
data(gradesTV)
```

Format

A data frame with 25 observations on the following 2 variables.

`TV` Number of hours per week students watch TV.

`Grades` Grades students got in a statistics class (out of 100).

Details

There are a few potential outliers in this data set. When analyzing the data one should consider how (if at all) these outliers may affect the estimates of correlation coefficient and regression parameters.

Source

Simulated data

Examples

```
data(gradesTV)
str(gradesTV)

plot(gradesTV)
makeTube(gradesTV$TV, gradesTV$Grades, 1.5, type='robust', homosk=FALSE)

lmPlot(gradesTV$TV, gradesTV$Grades, xAxis=4, xlab='time watching TV',
yR=0.2, highlight=c(1,15,20))
```

heartTr *Heart Transplant Data*

Description

The Stanford University Heart Transplant Study was conducted to determine whether an experimental heart transplant program increased lifespan. Each patient entering the program was designated officially a heart transplant candidate, meaning that he was gravely ill and would most likely benefit from a new heart. Then the actual heart transplant occurs between a few weeks to several months depending on the availability of a donor. Very few candidates during this waiting period show improvement and get *deselected* as a heart transplant candidate, but for the purposes of this experiment those patients were kept in the data as continuing candidates.

Usage

```
data(heartTr)
```

Format

A data frame with 103 observations on the following 8 variables.

`id` ID number of the patient.

`acceptyear` Year of acceptance as a heart transplant candidate.

`age` Age of the patient at the beginning of the study.

`survived` Survival status with levels `alive` and `dead`.

`survtime` Number of days patients were alive after the date they were determined to be a candidate for a heart transplant until the termination date of the study

`prior` Whether or not the patient had prior surgery with levels `yes` and `no`.

`transplant` Transplant status with levels `control` (did not receive a transplant) and `treatment` (received a transplant).

`wait` Waiting Time for Transplant

Source

http://www.stat.ucla.edu/~jsanchez/data/stanford.txt

References

Turnbull B, Brown B, and Hu M (1974). "Survivorship of heart transplant data." Journal of the American Statistical Association, vol. 69, pp. 74-80.

Examples

```
data(heartTr)
str(heartTr)
boxPlot(heartTr$survtime, heartTr$transplant,
ylab = 'Survival Time (days)')
mosaicplot(~ transplant + survived, data = heartTr)
```

`histPlot` *Histogram or hollow histogram*

Description

Create histograms and hollow histograms. This function permits easy color and appearance customization.

Usage

```
histPlot(x, col = fadeColor("black", "22"), border = "black", breaks = "default", probability = FALSE, h
```

Arguments

`x`	Numerical vector or a frequency table (matrix) where the first column represents the observed values and the second column the frequencies. See also `freqTable` argument.
`col`	Shading of the histogram bins.
`border`	Color of histogram bin borders.
`breaks`	A vector for the bin boundaries or an approximate number of bins.
`probability`	If `FALSE`, the frequency is plotted. If `TRUE`, then a probability density.
`hollow`	If `TRUE`, a hollow histogram will be created.
`add`	If `TRUE`, the histogram is added to the plot.
`lty`	Line type. Applies only if `hollow=TRUE`.
`lwd`	Line width. Applies only if `hollow=TRUE`.
`freqTable`	Set to `TRUE` if `x` is a frequency table.
`right`	Set to `FALSE` to assign values of `x` that fall on a bin margin to the left bin. Otherwise the ties default to the right bin.
`axes`	If `FALSE`, the axes are not plotted.
`xlab`	Label for the x axis.
`ylab`	Label for the y axis.
`xlim`	Limits for the x axis.
`ylim`	Limits for the y axis.
`...`	Additional arguments to `plot`. If `add` is `TRUE`, these arguments are ignored.

Author(s)

David Diez

See Also

`boxPlot`, `dotPlot`, `densityPlot`

Examples

```
data(run10)
par(mfrow=c(2,2))
histPlot(run10$time)
histPlot(run10$time[run10$gender=='M'], probability=TRUE, xlim=c(30, 180),
ylim=c(0, 0.025), hollow=TRUE)
histPlot(run10$time[run10$gender=='F'], probability=TRUE, add=TRUE,
hollow=TRUE, lty=3, border='red')
legend('topleft', col=c('black', 'red'), lty=2:3, legend=c('M','F'))
histPlot(run10$time, col=fadeColor('yellow', '33'), border='darkblue',
probability=TRUE, breaks=30, lwd=3)
brks <- c(40, 50, 60, 65, 70, 75, 80, seq(82.5, 120, 2.5), 125,
130, 135, 140, 150, 160, 170)
histPlot(run10$time, probability=TRUE, breaks=brks,
col=fadeColor('darkgoldenrod4', '33'))
```

---

house *United States House of Representatives historical make-up*

Description

The make-up of the United States House of Representatives every two years since 1789. The last Congress included is the 112th Congress, which completes its term in 2013.

Usage

```
data(house)
```

Format

A data frame with 112 observations on the following 12 variables.

`congress` The number of that year's Congress

`yearStart` Starting year

`yearEnd` Ending year

`seats` Total number of seats

`p1` Name of the first political party

`np1` Number of seats held by the first political party

`p2` Name of the second political party

`np2` Number of seats held by the second political party

`other` Other

`vac` Vacancy

`del` Delegate

`res` Resident commissioner

Source

Office of the Clerk of the U.S. House of Representatives Party Divisions:

http://clerk.house.gov/art_history/house_history/partyDiv.html

Data for Congresses 1-111 was recorded from the website above on November 1st, 2010. It appears this page was later moved to

http://artandhistory.house.gov/house_history/partyDiv.aspx

where data for Congress 112 was recorded on April 21, 2011.

Examples

```
data(house)

#=====> Examine two-party relationship since 1855 <=====#
these <- 34:112
COL   <- c("#EEDDBB", "#DDEEBB", "#DDDDDD",
           "#BBDDEE", "#EEE5E5", "#EECCCC")
party <- c("#2222FF", "#FF2222")
par(las=1)
plot(house$yearStart[these], 100*house$np1[these]/house$seats[these],
      type="n", xlab="Year", ylab="Percent of House seats", ylim=c(11, 93))
rect(1861.3, -1000, 1865.3, 1000, col=COL[1], border="#FFFFFF")
rect(1914.5, -1000, 1918.9, 1000, col=COL[2], border="#FFFFFF")
rect(1929, -1000, 1939, 1000, col=COL[3], border="#FFFFFF")
rect(1939.7, -1000, 1945.6, 1000, col=COL[4], border="#FFFFFF")
rect(1955.8, -1000, 1965.3, 1000, col=COL[5], border="#E2E2E2")
rect(1965.3, -1000, 1975.4, 1000, col=COL[6], border="#E2E2E2")
lines(house$yearStart[these], 100*house$np1[these]/house$seats[these],
      col=party[1])
lines(house$yearStart[these], 100*house$np2[these]/house$seats[these],
      col=party[2])
legend("topleft", lty=c(1,1), col=party,
      c("Democrats", "Republicans"), bg="#FFFFFF")
legend("topright", fill=COL,
      c("Civil War", "World War I", "Great Depression", "World War II",
        "Vietnam War Start", "Vietnam War Escalated"),
      bg="#FFFFFF", border="#FFFFFF")
```

---

houseRace10 *Election results for the 2010 U.S. House of Represenatives races*

Description

Election results for the 2010 U.S. House of Represenatives races

Usage

```
data(houseRace10)
```

Format

A data frame with 435 observations on the following 24 variables.

`id` Unique identifier for the race, which does not overlap with other 2010 races (see `govRace10` and `senateRace10`)

`state` State name

`abbr` State name abbreviation

`num` District number for the state

`name1` Name of the winning candidate

`perc1` Percentage of vote for winning candidate (if more than one candidate)

`party1` Party of winning candidate

`votes1` Number of votes for winning candidate

`name2` Name of candidate with second most votes

`perc2` Percentage of vote for candidate who came in second

`party2` Party of candidate with second most votes

`votes2` Number of votes for candidate who came in second

`name3` Name of candidate with third most votes

`perc3` Percentage of vote for candidate who came in third

`party3` Party of candidate with third most votes

`votes3` Number of votes for candidate who came in third

`name4` Name of candidate with fourth most votes

`perc4` Percentage of vote for candidate who came in fourth

`party4` Party of candidate with fourth most votes

`votes4` Number of votes for candidate who came in fourth

`name5` Name of candidate with fifth most votes

`perc5` Percentage of vote for candidate who came in fifth

`party5` Party of candidate with fifth most votes

`votes5` Number of votes for candidate who came in fifth

Details

This analysis in the Examples section was inspired by and is similar to that of Nate Silver's district-level analysis on the FiveThirtyEight blog in the New York Times:

http://fivethirtyeight.blogs.nytimes.com/2010/11/08/2010-an-aligning-election/

Source

Data was collected from MSNBC.com on November 9th, 2010.

Examples

```
data(houseRace10)
hr <- table(houseRace10[,c("abbr", "party1")])
nr <- apply(hr, 1, sum)

data(prRace08)
pr   <- prRace08[prRace08$state != "DC",c("state", "pObama")]
hr   <- hr[as.character(pr$state),]
(fit <- glm(hr ~ pr$pObama, family=binomial))

x1 <- pr$pObama[match(houseRace10$abbr, pr$state)]
y1 <- (houseRace10$party1 == "Democrat")+0
g  <- glm(y1 ~ x1, family=binomial)


x  <- pr$pObama[pr$state != "DC"]
nr <- apply(hr, 1, sum)
plot(x, hr[,"Democrat"]/nr, pch=19, cex=sqrt(nr), col="#22558844", xlim=c(20, 80), ylim=c(0, 1), xlab="Percent vot
X  <- seq(0, 100, 0.1)
lo <- -5.6079 + 0.1009*X
p  <- exp(lo)/(1+exp(lo))
lines(X, p)
abline(h=0:1, lty=2, col="#888888")
```

---

ipod *Length of songs on an iPod*

Description

A simulated data set on lengths of songs on an iPod.

Usage

```
data(ipod)
```

Format

A data frame with 3000 observations on the following variable.

`songLength` Length of song (in minutes).

Source

Simulated data.

Examples

```
data(ipod)
histPlot(ipod$songLength)
```

lmPlot *Linear regression plot with residual plot*

Description

Plot data, the linear model, and a residual plot simultaneously.

Usage

```
lmPlot(x, y, xAxis = 0, yAxis = 4, resAxis = 3, resSymm = TRUE,
wBox = TRUE, wLine = TRUE, lCol = "#00000088", lty = 1,
lwd = 1, xlab = "", ylab = "", marRes = NULL,
col = "#22558888", pch = 20, cex = 1.5, xR = 0.02,
yR = 0.1, xlim = NULL, ylim = NULL, subset = NULL,
parCustom = FALSE, myHeight = c(1, 0.45),
plots = c("both", "mainOnly", "resOnly"), highlight = NULL,
hlCol = NULL, hlCex = 1.5, hlPch = 20, na.rm=TRUE, ...)
```

Arguments

x	The x coordinates of points in the plot.
y	The y coordinates of points in the plot.
xAxis	The maximum number of x axis labels.
yAxis	The maximum number of y axis labels.
resAxis	The maximum number of y axis labels in the residual plot.
resSymm	Boolean determining whether the range of the residual plot should be symmetric about zero.
wBox	Boolean determining whether a box should be added around each plot.
wLine	Boolean determining whether to add a regression line to the plot.
lCol	The color of the regression line to be added.
lty	The line type of the regression line to be added.
lwd	The line width of the regression line to be added.
xlab	A label for the x axis.
ylab	A label for the y axis
marRes	Margin specified for the residuals.
col	Color of points.
pch	Plotting character.
cex	Plotting character size.
xR	Scaling the limits of the x axis. Ignored if `xlim` specified.
yR	Scaling the limits of the y axis. Ignored if `ylim` specified.
xlim	Limits for the x axis.

ylim	Limits for the y axis.
subset	A subset of the data to be used for the linear model.
parCustom	If TRUE, then the plotting margins are not modified automatically. This value should also be TRUE if the plots are being placed within a plot of multiple panels.
myHeight	A numerical vector of length 2 representing the ratio of the primary plot to the residual plot, in height.
plots	Not currently utilized.
highlight	Numerical vector specifying particular points to highlight.
hlCol	Color of highlighted points.
hlCex	Size of highlighted points.
hlPch	Plotting characters of highlighted points.
na.rm	Remove cases with NA values.
...	Additional arguments to plot.

Author(s)

David M Diez <david.m.diez@gmail.com>

See Also

makeTube

Examples

```
data(satGPA)
lmPlot(satGPA$SATSum, satGPA$FYGPA)

data(gradesTV)
lmPlot(gradesTV$TV, gradesTV$Grades, xAxis=4,
xlab='time watching TV', yR=0.2, highlight=c(1,15,20))
```

---

loop *Output a message while inside a loop*

Description

Output a message while inside a for loop to update the user on progress. This function is useful in tracking progress when the number of iterations is large or the procedures in each iteration take a long time.

Usage

```
loop(i, n = NULL, every = 1, extra=NULL)
```

Arguments

i	The index value used in the loop.
n	The last entry in the loop.
every	The number of loops between messages.
extra	Additional information to print.

Author(s)

David M Diez

See Also

myPDF

Examples

```
for(i in 1:160){
loop(i, 160, 20, paste("iter", i))
}
```

---

makeTube *Regression tube*

Description

Produce a linear, quadratic, or nonparametric tube for regression data.

Usage

```
makeTube(x, y, Z=2, R=1, col='#00000022', border='#00000000',
  type=c('lin', 'quad', 'robust'), stDev=c('constant', 'linear', 'other'),
  length.out=99, bw='default', plotTube=TRUE, addLine=TRUE, ...)
```

Arguments

x	x coordinates.
y	y coordinates.
Z	Number of standard deviations out from the regression line to extend the tube.
R	Control of how far the tube extends to the left and right.
col	Fill color of the tube.
border	Border color of the tube.
type	The type of model fit to the data. Here 'robust' results in a nonparametric estimate.

stDev	Choices are constant variance ('constant'), the standard deviation of the errors changes linearly ('linear'), or the standard deviation of the errors should be estimated using nonparametric methods ('other').
length.out	The number of observations used to build the regression model. This argument may be increased to increase the smoothing of a quadratic or nonparametric curve.
bw	Bandwidth used if type='robust' or homosk=FALSE.
plotTube	Whether the tube should be plotted.
addLine	Whether the linear model should be plotted.
...	Additional arguments passed to the lines function if addLine=TRUE.

Value

X	x coordinates for the regression model.
Y	y coordinates for the regression model.
tubeX	x coordinates for the boundary of the tube.
tubeY	y coordinates for the boundary of the tube.

Author(s)

David M Diez

See Also

lmPlot

Examples

```
#===> possum example <===#
data(possum)
x <- possum$totalL
y <- possum$headL
plot(x,y)
makeTube(x,y,1)
makeTube(x,y,2)
makeTube(x,y,3)

#===> Grades and TV example <===#
data(gradesTV)
par(mfrow=c(2,2))
plot(gradesTV)
makeTube(gradesTV$TV, gradesTV$Grades, 1.5)
plot(gradesTV)
makeTube(gradesTV$TV, gradesTV$Grades, 1.5, stDev='o')
plot(gradesTV)
makeTube(gradesTV$TV, gradesTV$Grades, 1.5, type='robust')
plot(gradesTV)
makeTube(gradesTV$TV, gradesTV$Grades, 1.5, type='robust', stDev='o')
```

```
#===> What can go wrong with a basic least squares model <===#
par(mfrow=c(1,3), mar=c(2.5, 2.5, 1, 2.5))
# 1
x <- runif(100)
y <- 25*x-20*x^2+rnorm(length(x), sd=1.5)
plot(x,y)
makeTube(x,y,type='q')
# 2
x <- c(-0.6, -0.46, -0.091, runif(97))
y <- 25*x + rnorm(length(x))
y[2] <- y[2] + 8
y[1] <- y[1] + 1
plot(x,y,ylim=range(y)+c(-10,5))
makeTube(x,y)
# 3
x <- runif(100)
y <- 5*x + rnorm(length(x), sd=x)
plot(x,y)
makeTube(x, y, stDev='l', bw=0.03)
```

`mammals` *Sleep in Mammals*

Description

This data set includes data for 39 species of mammals distributed over 13 orders. The data were used for analyzing the relationship between constitutional and ecological factors and sleeping in mammals. Two qualitatively different sleep variables (dreaming and non dreaming) were recorded. Constitutional variables such as life span, body weight, brain weight and gestation time were evaluated. Ecological variables such as severity of predation, safety of sleeping place and overall danger were inferred from field observations in the literature.

Usage

```
data(mammals)
```

Format

A data frame with 62 observations on the following 11 variables.

`Species` Species of mammals

`BodyWt` Total body weight of the mammal (in kg)

`BrainWt` Brain weight of the mammal (in kg)

`NonDreaming` Number of hours of non dreaming sleep

`Dreaming` Number of hours of dreaming sleep

`TotalSleep` Total number of hours of sleep

`LifeSpan` Life span (in years)

`Gestation` Gestation time (in days)

`Predation` An index of how likely the mammal is to be preyed upon. 1 = least likely to be preyed upon. 5 = most likely to be preyed upon.

`Exposure` An index of the how exposed the mammal is during sleep. 1 = least exposed (e.g., sleeps in a well-protected den). 5 = most exposed.

`Danger` An index of how much danger the mammal faces from other animals. This index is based upon Predation and Exposure. 1 = least danger from other animals. 5 = most danger from other animals.

Source

http://www.statsci.org/data/general/sleep.txt

References

T. Allison and D. Cicchetti, "Sleep in mammals: ecological and constitutional correlates," Arch. Hydrobiol, vol. 75, p. 442, 1975.

Examples

```
data(mammals)
lmPlot(log(mammals$BodyWt), log(mammals$BrainWt))
```

---

marathon *New York City Marathon Times*

Description

Marathon times of male and female winners of the New York City Marathon 1970-1999.

Usage

```
data(marathon)
```

Format

A data frame with 60 observations on the following 3 variables.

`Year` Year

`Gender` Gender

`Time` Running time (in hours)

Source

http://www.webcitation.org/5kx7ilFLp

Examples

```
data(marathon)
str(marathon)
histPlot(marathon$Time)
boxPlot(marathon$Time, horiz = TRUE, fact = marathon$Gender)
```

`marioKart` *Wii Mario Kart auctions from Ebay*

Description

Auction data from Ebay for the game Mario Kart for the Nintendo Wii. This data was collected in early October, 2009.

Usage

```
data(marioKart)
```

Format

A data frame with 143 observations on the following 12 variables. All prices are in US dollars.

`ID` Auction ID assigned by Ebay.

`duration` Auction length, in days.

`nBids` Number of bids.

`cond` Game condition, either `new` or `used`.

`startPr` Start price of the auction.

`shipPr` Shipping price.

`totalPr` Total price, which equals the auction price plus the shipping price.

`shipSp` Shipping speed or method.

`sellerRate` The seller's rating on Ebay. This is the number of positive ratings minus the number of negative ratings for the seller.

`stockPhoto` Whether the auction feature photo was a stock photo or not. If the picture was used in many auctions, then it was called a stock photo.

`wheels` Number of Wii wheels included in the auction. These are steering wheel attachments to make it seem as though you are actually driving in the game. When used with the controller, turning the wheel actually causes the character on screen to turn.

`title` The title of the auctions.

Details

There are several interesting features in the data. First off, note that there are two outliers in the data. These serve as a nice example of what one should do when encountering an outlier: examine the data point and remove it only if there is a good reason. In these two cases, we can see from the auction titles that they included other items in their auctions besides the game, which justifies removing them from the data set.

This data set includes all auctions for a full week in October, 2009. Auctions were included in the data set if they satisfied a number of conditions. (1) They were included in a search for "wii mario kart" on ebay.com, (2) items were in the Video Games > Games > Nintendo Wii section of Ebay, (3) the listing was an auction and not exclusively a "Buy it Now" listing (sellers sometimes offer an optional higher price for a buyer to end bidding and win the auction immediately, which is an *optional* Buy it Now auction), (4) the item listed was the actual game, (5) the item was being sold from the US, (6) the item had at least one bidder, (7) there were no other items included in the auction with the exception of racing wheels, either generic or brand-name being acceptable, and (8) the auction did not end with a Buy It Now option.

References

http://www.ebay.com/

http://www.openintro.org/

Examples

```
data(marioKart)

#===> Identify the outliers <===#
boxPlot(marioKart$totalPr, marioKart$cond, horiz=TRUE)
toss <- which(marioKart$totalPr > 80)
lines(rep(marioKart$totalPr[toss[1]], 2), c(2.4, 2))
text(marioKart$totalPr[toss[1]], 2.4, marioKart$title[toss[1]],
pos=3, cex=0.5)
lines(rep(marioKart$totalPr[toss[2]], 2), c(1.6, 2))
text(marioKart$totalPr[toss[2]], 1.6, marioKart$title[toss[2]],
pos=1, cex=0.5)
marioKart[toss, ]
# the other two points marked on the boxplot are legitimate auctions

#===> Replot without the outliers <===#
boxPlot(marioKart$totalPr[-toss], marioKart$cond[-toss], horiz=TRUE)

#===> Fit a Multiple Regression Model <===#
mk <- marioKart[-toss,]
summary(lm(totalPr ~ cond + stockPhoto + duration + wheels, mk))
summary(lm(totalPr ~ cond + stockPhoto + wheels, mk))
summary(fit <- lm(totalPr ~ cond + wheels, mk))

#===> Fit Diagnostics <===#
e <- fit$res
f <- fit$fit
par(mfrow=c(2,3), mar=c(4, 4, 2, 1))
```

```
qqnorm(e, ylab="Residuals", main="")
plot(e, xlab="Order of collection", ylab="Residuals")
plot(f, e, xlab="Fitted values", ylab="Residuals")
plot(f, (abs(e)), xlab="Fitted values",
     ylab="Absolute value of residuals")
boxPlot(e, mk$cond, xlab="Condition", ylab="Residuals")
plot(mk$wheels, e, xlab="Number of wheels", ylab="Residuals",
     main="Notice curvature")
```

MLB *Salary data for Major League Baseball (2010)*

Description

Salary data for Major League Baseball players in the year 2010.

Usage

```
data(MLB)
```

Format

A data frame with 828 observations on the following 4 variables.

`player` Player name

`team` Team

`position` Field position

`salary` Salary (in $1000s)

Source

Collected from the following page (and its linked pages) on February 23rd, 2011:

http://content.usatoday.com/sportsdata/baseball/mlb/salaries/team

Examples

```
data(MLB)

#=====> Basic Histogram <=====#
hist(MLB$salary/1000, main="", breaks=15, xlab="Salary (millions of dollars)", axes=FALSE, ylab="", col="#2255884
axis(1, seq(0, 40, 10))
axis(2, c(0, 500))
axis(2, seq(100, 400, 100), rep("", 4), tcl=-0.2)

#=====> Histogram on Log Scale <=====#
hist(log(MLB$salary/1000), main="", breaks=15, xlab="log(Salary)", axes=FALSE, ylab="", col="#22558844")
axis(1) #, seq(0, 40, 10))
axis(2, seq(0, 300, 100))
```

```
#=====> Box plot of log(salary) against position <=====#
par(las=1, mar=c(4, 8, 1, 1))
boxPlot(log(MLB$salary/1000), MLB$position, horiz=TRUE, ylab="")
```

`mlbBat10` *Major League Baseball Player Hitting Statistics for 2010*

Description

Major League Baseball Player Hitting Statistics for 2010.

Usage

```
data(mlbBat10)
```

Format

A data frame with 1199 observations on the following 19 variables.

`name` Player name

`team` Team abbreviation

`position` Player position

`G` Number of games

`AB` Number of at bats

`R` Number of runs

`H` Number of hits

`2B` Number of doubles

`3B` Number of triples

`HR` Number of home runs

`RBI` Number of runs batted in

`TB` Total bases, computed as 3*HR + 2*3B + 1*2B + H

`BB` Number of walks

`SO` Number of strikeouts

`SB` Number of stolen bases

`CS` Number of times caught stealing

`OBP` On base percentage

`SLG` Slugging percentage (TB / AB)

`AVG` Batting average

Source

Data was collected from MLB.com on April 22nd, 2011.

Examples

```
data(mlbBat10)
d   <- mlbBat10[mlbBat10$AB > 200,]
pos <- list(c("OF"), c("1B", "2B", "3B", "SS"), "DH", "C")
POS <- c("OF", "IF", "DH", "C")

#=====> On-base Percentage Across Positions <=====#
out <- c()
gp  <- c()
for(i in 1:length(pos)){
these <- which(d$pos %in% pos[[i]])
out   <- c(out, d[these,"OBP"])
gp    <- c(gp, rep(POS[i], length(these)))
}
plot(out ~ as.factor(gp))
summary(lm(out ~ as.factor(gp)))
anova(lm(out ~ as.factor(gp)))

#=====> Batting Average Across Positions <=====#
out <- c()
gp  <- c()
for(i in 1:length(pos)){
these <- which(d$pos %in% pos[[i]])
out   <- c(out, d[these,"AVG"])
gp    <- c(gp, rep(POS[i], length(these)))
}
plot(out ~ as.factor(gp))
summary(lm(out ~ as.factor(gp)))
anova(lm(out ~ as.factor(gp)))

#=====> Home Runs Across Positions <=====#
out <- c()
gp  <- c()
for(i in 1:length(pos)){
these <- which(d$pos %in% pos[[i]])
out   <- c(out, d[these,"HR"])
gp    <- c(gp, rep(POS[i], length(these)))
}
plot(out ~ as.factor(gp))
summary(lm(out ~ as.factor(gp)))
anova(lm(out ~ as.factor(gp)))

#=====> Runs Batted In Across Positions <=====#
out <- c()
gp  <- c()
for(i in 1:length(pos)){
these <- which(d$pos %in% pos[[i]])
out   <- c(out, d[these,"RBI"])
gp    <- c(gp, rep(POS[i], length(these)))
}
plot(out ~ as.factor(gp))
summary(lm(out ~ as.factor(gp)))
```

```
anova(lm(out ~ as.factor(gp)))
```

---

myPDF *Custom PDF function*

Description

A similar function to pdf and png, except that different defaults are provided, including for the plotting parameters.

Usage

```
myPDF(fileName, width = 5, height = 3,
      mar = c(3.9, 3.9, 1, 1),
      mgp = c(2.8, 0.55, 0),
      las = 1, tcl=-0.3, ...)

myPNG(fileName, width = 600, height = 400,
      mar = c(3.9, 3.9, 1, 1),
      mgp = c(2.8, 0.55, 0),
      las = 1, tcl=-0.3, ...)
```

Arguments

fileName	File name for the image to be output. The name should end in .pdf.
width	The width of the image file (inches). Default: 5.
height	The height of the image file (inches). Default: 3.
mar	Plotting margins. To change, input a numerical vector of length 4.
mgp	Margin graphing parameters. To change, input a numerical vector of length 3. The first argument specifies where x and y labels are placed; the second specifies the axis labels are placed; and the third specifies how far to pull the entire axis from the plot.
las	Orientation of axis labels. Input 0 for the default.
tcl	The tick mark length as a proportion of text height. The default is -0.5.
...	Additional arguments to par.

Author(s)

David M Diez

See Also

edaPlot

Examples

```
data(marioKart)
#=====> Save a plot to a PDF <=====#
# myPDF("myPlot.pdf")
data(run10)
histPlot(run10$time)
# dev.off()

#=====> Save a plot to a PNG <=====#
# myPNG("myPlot.pdf")
data(run10)
histPlot(run10$time)
# dev.off()
```

`normTail` *Normal distribution tails*

Description

Produce a normal (or t) distribution and shaded tail.

Usage

```
normTail(m = 0, s = 1, L = NULL, U = NULL, M = NULL, df=1000,
curveColor=1, border = 1, col = "#CCCCCC", xlim = NULL,
ylim=NULL, xlab = "", ylab = "", digits = 2, axes = 1,
detail = 999, xLab = c("number", "symbol"), cex.axis = 1,
xAxisIncr=1, ...)
```

Arguments

`m`	Numerical value for the distribution mean.
`s`	Numerical value for the distribution standard deviation.
`L`	Numerical value representing the cutoff for a shaded lower tail.
`U`	Numerical value representing the cutoff for a shaded upper tail.
`M`	Numerical value representing the cutoff for a shaded central region.
`df`	Numerical value describing the degrees of freedom. Default is `1000`, which results in a nearly normal distribution. Small values may be useful to emphasize small tails.
`curveColor`	The color for the distribution curve.
`border`	The color for the border of the shaded area.
`col`	The color for filling the shaded area.
`xlim`	Limits for the x axis.
`ylim`	Limits for the y axis.

xlab	A title for the x axis.
ylab	A title for the y axis.
digits	The maximum number of digits past the decimal to use in axes values.
axes	A numeric value denoting whether to draw both axes (3), only the vertical axes (2), only the horizontal axes (1, the default), or no axes (0).
detail	A number describing the number of points to use in drawing the normal curve. Smaller values correspond to a less smooth curve but reduced memory usage in the final file.
xLab	If "number", then the axis is drawn at the mean, and every standard deviation out until the third standard deviation. If "symbol", then Greek letters are used for standard deviations from three standard deviations from the mean.
cex.axis	Numerical value controlling the size of the axis labels.
xAxisIncr	A number describing how often axis labels are placed, scaled by standard deviations. This argument is ignored if xLab="symbol".
...	Additional arguments to plot.

Author(s)

DM Diez

See Also

buildAxis

Examples

```
par(mfrow=c(2,3), mar=c(3,3,1,1))
normTail(3,2,5)
normTail(3,2,1, xLab='symbol')
normTail(3,2,M=1:2, xLab='symbol', cex.axis=0.8)
normTail(3,2,U=5,axes=FALSE)
normTail(L=-1, U=2, M=c(0,1), axes=3, xAxisIncr=2)
normTail(L=-1, U=2, M=c(0,1), xLab='symbol', cex.axis=0.8, xAxisIncr=2)
```

---

possum *possum*

Description

Data representing possums in Australia and New Guinea. This is a copy of the data set by the same name in the DAAG package, however, the data set included here includes fewer variables.

Usage

```
data(possum)
```

Format

A data frame with 104 observations on the following 8 variables.

`site` The site number where the possum was trapped.

`pop` Population, either `Vic` (Victoria) or`other` (New South Wales or Queensland).

`sex` Gender, either `m` (male) or `f` (female).

`age` Age.

`headL` Head length, in mm.

`skullW` Skull width, in mm.

`totalL` Total length, in cm.

`tailL` Tail length, in cm.

Source

Lindenmayer, D. B., Viggers, K. L., Cunningham, R. B., and Donnelly, C. F. 1995. Morphological variation among columns of the mountain brushtail possum, Trichosurus caninus Ogilby (Phalangeridae: Marsupiala). Australian Journal of Zoology 43: 449-458.

References

http://www.openintro.org/

Examples

```
data(possum)
par(mfrow=1:2)
plot(possum$headL, possum$skullW)
densityPlot(possum$totalL, possum$sex, key=c('f','m'),
xlab='total length (cm)')
legend('topright', col=c('black', 'red'), lty=1:2, legend=c('f', 'm'))
```

---

president *United States Presidental History*

Description

Summary of the changes in the president and vice president for the United States of America.

Usage

```
data(president)
```

Format

A data frame with 67 observations on the following 5 variables.

`potus` President of the United States

`party` Political party of the president

`start` Start year

`end` End year

`vpotus` Vice President of the United States

Source

Presidents of the United States (table) – infoplease.com (visited: Nov 2nd, 2010)

http://www.infoplease.com/ce6/history/A0840075.html

Examples

```
data(president)
```

---

prRace08 *Election results for the 2008 U.S. Presidential race*

Description

Election results for the 2008 U.S. Presidential race

Usage

```
data(prRace08)
```

Format

A data frame with 51 observations on the following 7 variables.

`state` State name abbreviation

`stateFull` Full state name

`nObama` Number of votes for Barack Obama

`pObama` Proportion of votes for Barack Obama

`nMcCain` Number of votes for John McCain

`pMcCain` Proportion of votes for John McCain

`elVotes` Number of electoral votes for a state

Details

In Nebraska, 4 electoral votes went to McCain and 1 to Obama. Otherwise the electoral votes were a winner-take-all.

Source

Presidential Election of 2008, Electoral and Popular Vote Summary, collected on April 21, 2011 from

http://www.infoplease.com/us/government/presidential-election-vote-summary.html

Examples

```
#===> Obtain 2010 US House Election Data <===#
data(houseRace10)
hr <- table(houseRace10[,c("abbr", "party1")])
nr <- apply(hr, 1, sum)

#===> Obtain 2008 President Election Data <===#
data(prRace08)
pr   <- prRace08[prRace08$state != "DC",c("state", "pObama")]
hr   <- hr[as.character(pr$state),]
(fit <- glm(hr ~ pr$pObama, family=binomial))

#===> Visualizing Binomial outcomes <===#
x  <- pr$pObama[pr$state != "DC"]
nr <- apply(hr, 1, sum)
plot(x, hr[,"Democrat"]/nr, pch=19, cex=sqrt(nr), col="#22558844",
    xlim=c(20, 80), ylim=c(0, 1), xlab="Percent vote for Obama in 2008",
    ylab="Probability of Democrat winning House seat")

#===> Logistic Regression <===#
x1 <- pr$pObama[match(houseRace10$abbr, pr$state)]
y1 <- (houseRace10$party1 == "Democrat")+0
g  <- glm(y1 ~ x1, family=binomial)
X  <- seq(0, 100, 0.1)
lo <- -5.6079 + 0.1009*X
p  <- exp(lo)/(1+exp(lo))
lines(X, p)
abline(h=0:1, lty=2, col="#888888")
```

`run10` *Cherry Blossom 10 mile run data, 2009*

Description

14 variables for all 14974 10 mile participants in the 2009 Cherry Blossom Run.

Usage

```
data(run10)
```

Format

A data frame with 14974 observations on the following 14 variables.

`place` Finishing position. Separate positions are provided for each gender.

`time` The total run time.

`netTime` The run time from the start line to the finish line.

`pace` The listed pace for each runner.

`age` Age.

`gender` Gender.

`first` First name.

`last` Last name.

`city` Hometown city.

`state` Hometown state.

`country` Hometown country.

`div` Running division (age group).

`divPlace` Division place, also broken up by gender.

`divTot` Total number of people in the division (again, also split by gender).

Source

~~ http://www.cherryblossom.org/ ~~

References

~~ http://www.openintro.org/ ~~

Examples

```
data(run10)

#===> men's times <===#
histPlot(run10$time[run10$gender == 'M'])

#===> times by gender <===#
densityPlot(run10$time, run10$gender, key=c('M','F'))
legend('topright', lty=2:1, col=c('red','black'), legend=c('M','F'))
```

satGPA *SAT and GPA data*

Description

SAT and GPA data for 1000 students at an unnamed college.

Usage

```
data(satGPA)
```

Format

A data frame with 1000 observations on the following 6 variables.

`sex` Gender of the student.

`SATV` Verbal SAT percentile.

`SATM` Math SAT percentile.

`SATSum` Total of verbal and math SAT percentiles.

`HSGPA` High school grade point average.

`FYGPA` First year (college) grade point average.

Source

Educational Testing Service originally collected the data.

References

Data retrieved from

https://www.dartmouth.edu/~chance/course/Syllabi/Princeton96/Class12.html

Data utilized in Chapter 7 of the Open Intro Statistics book: http://www.openintro.org/

Examples

```
data(satGPA)

par(mfrow=2:1)

plot(satGPA$SATSum/2, satGPA$FYGPA)
g <- lm(satGPA$FYGPA ~ I(satGPA$SATSum/2))
summary(g)
abline(g)

plot(satGPA$SATM, satGPA$FYGPA)
g <- lm(satGPA$FYGPA ~ satGPA$SATM)
summary(g)
abline(g)
```

`senateRace10` *Election results for the 2010 U.S. Senate races*

Description

Election results for the 2010 U.S. Senate races

Usage

```
data(senateRace10)
```

Format

A data frame with 38 observations on the following 23 variables.

`id` Unique identifier for the race, which does not overlap with other 2010 races (see `govRace10` and `houseRace10`)

`state` State name

`abbr` State name abbreviation

`name1` Name of the winning candidate

`perc1` Percentage of vote for winning candidate (if more than one candidate)

`party1` Party of winning candidate

`votes1` Number of votes for winning candidate

`name2` Name of candidate with second most votes

`perc2` Percentage of vote for candidate who came in second

`party2` Party of candidate with second most votes

`votes2` Number of votes for candidate who came in second

`name3` Name of candidate with third most votes

`perc3` Percentage of vote for candidate who came in third

`party3` Party of candidate with third most votes

`votes3` Number of votes for candidate who came in third

`name4` Name of candidate with fourth most votes

`perc4` Percentage of vote for candidate who came in fourth

`party4` Party of candidate with fourth most votes

`votes4` Number of votes for candidate who came in fourth

`name5` Name of candidate with fifth most votes

`perc5` Percentage of vote for candidate who came in fifth

`party5` Party of candidate with fifth most votes

`votes5` Number of votes for candidate who came in fifth

Source

Data was collected from MSNBC.com on November 9th, 2010.

Examples

```
data(senateRace10)
table(senateRace10$party1)
histPlot(senateRace10$perc1, xlab="Winning candidate vote percentage")
```

---

smoking *UK Smoking Data*

Description

Survey data on smoking habits from the UK. The data set can be used for analyzing the demographic characteristics of smokers and types of tobacco consumed.

Usage

```
data(smoking)
```

Format

A data frame with 1691 observations on the following 12 variables.

`gender` Gender with levels `Female` and `Male`.

`age` Age.

`maritalStatus` Marital status with levels `Divorced`, `Married`, `Separated`, `Single` and `Widowed`.

`highestQualification` Highest education level with levels `A Levels`, `Degree`, `GCSE/CSE`, `GCSE/O Level`, `Higher/Sub Degree`, `No Qualification`, `ONC/BTEC` and `Other/Sub Degree`

`nationality` Nationality with levels `British`, `English`, `Irish`, `Scottish`, `Welsh`, `Other`, `Refused` and `Unknown`.

`ethnicity` Ethnicity with levels `Asian`, `Black`, `Chinese`, `Mixed`, `White` and `Refused Unknown`.

`grossIncome` Gross income with levels `Under 2,600`, `2,600 to 5,200`, `5,200 to 10,400`, `10,400 to 15,600`, `15,600 to 20,800`, `20,800 to 28,600`, `28,600 to 36,400`, `Above 36,400`, `Refused` and `Unknown`.

`region` Region with levels `London`, `Midlands & East Anglia`, `Scotland`, `South East`, `South West`, `The North` and `Wales`

`smoke` Smoking status with levels `No` and `Yes`

`amtWeekends` Number of cigarettes smoked per day on weekends.

`amtWeekdays` Number of cigarettes smoked per day on weekdays.

`type` Type of cigarettes smoked with levels `Packets`, `Hand-Rolled`, `Both/Mainly Packets` and `Both/Mainly Hand-Rolled`

Source

http://www.stats4schools.gov.uk/large_datasets/smoking/default.asp

Examples

```
data(smoking)
str(smoking)
histPlot(smoking$amtWeekends)
histPlot(smoking$amtWeekdays)
table(smoking$smoke, smoking$gender)
mosaicplot(~ smoke + maritalStatus, data = smoking)
barplot(sort(table(smoking$maritalStatus), decreasing = TRUE))
```

---

textbooks *Textbook data for UCLA Bookstore and Amazon*

Description

A random sample was taken of nearly 10% of UCLA courses. The most expensive textbook for each course was identified, and its new price at the UCLA Bookstore and on Amazon.com were recorded.

Usage

```
data(textbooks)
```

Format

A data frame with 73 observations on the following 7 variables.

`deptAbbr` Course department (abbreviated).

`course` Course number.

`ibsn` Book ISBN.

`uclaNew` New price at the UCLA Bookstore.

`amazNew` New price on Amazon.com.

`more` Whether additional books were required for the course (`Y` means "yes, additional books were required").

`diff` The UCLA Bookstore price minus the Amazon.com price for each book.

Details

The sample represents only courses where textbooks were listed online through UCLA Bookstore's website. The most expensive textbook was selected based on the UCLA Bookstore price, which may insert bias into the data; for this reason, it may be beneficial to analyze only the data where `more` is `"N"`.

Source

This data was collected by David M Diez on April 24th.

References

See Section 5.1 of the Open Intro Statistics textbook: http://www.openintro.org/

Examples

```
data(textbooks)
#===> an improper analysis <===#
boxPlot(textbooks$uclaNew, xlim=c(0.5,2.5))
boxPlot(textbooks$amazNew, add=2)
axis(1, at=1:2, labels=c('UCLA Bookstore', 'Amazon'))
t.test(textbooks$uclaNew, textbooks$amazNew)

#===> a reasonable analysis <===#
#     the differences are moderately skewed
#     the sample size is sufficiently large to justify t test
histPlot(textbooks$diff)
t.test(textbooks$diff)
```

---

tips *Tip data*

Description

A simulated data set of tips over a few weeks on a couple days per week. Each tip is associated with a single group, which may include several bills and tables (i.e. groups paid in one lump sum in simulations).

Usage

```
data(tips)
```

Format

A data frame with 95 observations on the following 5 variables.

`week` Week number.

`day` Day, either `Friday` or `Tuesday`.

`nPeop` Number of people associated with the group.

`bill` Total bill for the group.

`tip` Total tip from the group.

Details

This data set was built using simulations of tables, then bills, then tips based on the bills. Large groups were assumed to only pay the gratuity, which is evident in the data. Tips were set to be plausible round values; they were often (but not always) rounded to dollars, quarters, etc.

Source

Simulated data set.

References

http://www.openintro.org/

Examples

```
data(tips)
par(mfrow=c(2,2))
boxPlot(tips$tip, tips$day)
densityPlot(tips$tip, tips$week, key=1:3)
legend('topright', lty=1:3, col=c('black', 'red', 'blue'), legend=1:3)
dotPlot(tips$tip)
densityPlot(tips$tip, tips$day)
legend('topright', col=c('black','red'), lty=1:2,
legend=c('Tuesday', 'Friday'))
```

---

unempl *Annual unemployment since 1890*

Description

A compilation of two data sets that provides an estimate of unemployment from 1890 to 2010.

Usage

```
data(unempl)
```

Format

A data frame with 121 observations on the following 3 variables.

`year` Year

`unemp` Unemployment rate, in percent

`usData` `1` if from the Bureau of Labor Statistics, `0` otherwise

Source

The data are from Wikipedia at the following URL accessed on November 1st, 2010:

http://en.wikipedia.org/wiki/File:US_Unemployment_1890-2009.gif

Below is a direct quotation from Wikipedia describing the sources of the data:

Own work by Peace01234 Complete raw data are on Peace01234. 1930-2009 data are from Bureau of Labor Statistics, Employment status of the civilian noninstitutional population, 1940 to date ftp://ftp.bls.gov/pub/special.requests/lf/aat1.txt, retrieved March 6, 2009 and [1] retrieved February 12, 2010. Data prior to 1948 are for persons age 14 and over. Data beginning in 1948 are for persons age 16 and over. See also "Historical Comparability" under the Household Data section of the Explanatory Notes at http://www.bls.gov/cps/eetech_methods.pdf. 1890-1930 data are from Christina Romer (1986). "Spurious Volatility in Historical Unemployment Data", The Journal of Political Economy, 94(1): 1-37. 1930-1940 data are from Robert M. Coen (1973). "Labor Force and Unemployment in the 1920's and 1930's: A Re-Examination Based on Postwar Experience", The Review of Economics and Statistics, 55(1): 46-55. Unemployment data was only surveyed once each decade until 1940 when yearly surveys were begun. The yearly data estimates before 1940 are based on the decade surveys combined with other relevant surveys that were collected during those years. The methods are described in detail by Coen and Romer.

Examples

```
data(unempl)

#=====> Time Series Plot of Data <=====#
COL   <- c("#DDEEBB", "#EEDDBB", "#BBDDEE", "#FFD5DD", "#FFC5CC")
plot(unempl$year, unempl$unemp, type="n")
rect(0, -50, 3000, 100, col="#E2E2E2")
rect(1914.5, -1000, 1918.9, 1000, col=COL[1], border="#E2E2E2")
rect(1929, -1000, 1939, 1000, col=COL[2], border="#E2E2E2")
rect(1939.7, -1000, 1945.6, 1000, col=COL[3], border="#E2E2E2")
rect(1955.8, -1000, 1965.3, 1000, col=COL[4], border="#E2E2E2")
rect(1965.3, -1000, 1975.4, 1000, col=COL[5], border="#E2E2E2")
abline(h=seq(0,50,5), col="#F8F8F8", lwd=2)
abline(v=seq(1900, 2000, 20), col="#FFFFFF", lwd=1.3)
lines(unempl$year, unempl$unemp)
points(unempl$year, unempl$unemp, pch=20)
legend("topright", fill=COL,
      c("World War I", "Great Depression", "World War II",
        "Vietnam War Start", "Vietnam War Escalated"),
      bg="#FFFFFF", border="#FFFFFF")
```

Index