# Package: rafalib (via r-universe)

October 23, 2024

**Version** 1.0.0

**Title** Convenience Functions for Routine Data Exploration

**Description** A series of shortcuts for routine tasks originally developed by Rafael A. Irizarry to facilitate data exploration.

**Author** Rafael A. Irizarry and Michael I. Love

**Maintainer** Rafael A. Irizarry `<rafa@jimmy.harvard.edu>`

**Imports** RColorBrewer

**License** Artistic-2.0

**LazyData** yes

**NeedsCompilation** no

**Date/Publication** 2015-08-09 00:00:40

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

**RemoteUrl** https://github.com/cran/rafalib

**RemoteRef** HEAD

**RemoteSha** 2549c7361fcbe335819a4647ea873574a2779dde

## Contents

---

## `as.fumeric` *converts to factor and then numeric*

### Description

Converts a vector of characters into factors and then converts these into numeric.

### Usage

```
as.fumeric(x, levels = unique(x))
```

### Arguments

| | |
|---|---|
| `x` | a character vector |
| `levels` | the leves to be used in the call to factor |

### Author(s)

Rafael A. Irizarry

### Examples

```
group = c("a","a","b","b")
plot(seq_along(group),col=as.fumeric(group))
```

---

## `bartab` *bartab*

### Description

Plot the overlap of three groups with a barplot

### Usage

```
bartab(x, y, z, names, skipNone = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `x` | logical |
| `y` | logical |
| `z` | logical |
| `names` | a character vector of length 3 |
| `skipNone` | remove the "none" group |
| `...` | further arguments passed on to `barplot` |

## Author(s)

Michael I. Love

---

# `imagemat` *image of a matrix*

## Description

Produces an image of a matrix which matches the natural orientation.

## Usage

```
imagemat(x, col = colorRampPalette(c("white", "black"))(9), las = 1,
  xlab = "", ylab = "", ...)
```

## Arguments

| | |
|---|---|
| `x` | the matrix |
| `col` | the colors |
| `las` | as in par |
| `xlab` | x-axis title |
| `ylab` | y-axis title |
| `...` | arguments passed to image |

## Author(s)

Michael I. Love

## Examples

```
x <- matrix(c(1,0,0,0,1,
              1,1,0,1,1,
              1,0,1,0,1,
              1,0,0,0,1,
              1,0,0,0,1),
ncol=5,byrow=TRUE)

imagemat(x)
```

---

`imagesort` *image with sorted rows*

---

## Description

the rows are sorted such that the first column has 2 blocks, the second column has 4 blocks, etc. see example("imagesort")

## Usage

```
imagesort(x, col = c("white", "black"), ...)
```

## Arguments

| | |
|---|---|
| `x` | a matrix of 0s and 1s |
| `col` | the colors of 0 and 1 |
| `...` | arguments to heatmap |

## Author(s)

Michael I. Love

## Examples

```
x <- replicate(4,sample(0:1,40,TRUE))
imagesort(x)
```

---

`install_bioc` *Install or update Bioconductor and CRAN packages*

---

## Description

This is function simply a wrapper for `biocLite`. It first sources the code from the Bioconductor site then calls `biocLite`.

## Usage

```
install_bioc(...)
```

## Arguments

| | |
|---|---|
| `...` | arguments passed on to `biocLite` |

## Details

Note that once you run this function in a session, you no longer need to call since you can call `biocLite` directly.

## Author(s)

Rafael A. Irizarry

---

`largeobj` *What are the largest objects in memory?*

## Description

This function lists all the objects in the global environmnet and lists the `n` largest.

## Usage

```
largeobj(n = 5, units = "Mb")
```

## Arguments

| | |
|---|---|
| `n` | the number of objects to return |
| `units` | units to display, see `?object.size` |

## Value

a named character string of the size of the 'n' largest objects

## Author(s)

Michael I. Love

---

`maplot` *Bland Altman plot aka MA plot*

## Description

Takes two vectors x and y and plots M=y-x versus A=(x+y)/2. If the vectors a more longer than length n the data is sampled to size n. A smooth curve is added to show trends.

## Usage

```
maplot(x, y, n = 10000, subset = NULL, xlab = NULL, ylab = NULL,
  curve.add = TRUE, curve.col = 2, curve.span = 1/2, curve.lwd = 2,
  curve.n = 2000, ...)
```

## Arguments

| | |
|---|---|
| `x` | a numeric vector |
| `y` | a numeric vector |
| `n` | a numeric value. If `length(x)` is larger than `n`, the `x` and `y` are sampled down. |
| `subset` | index of the points to be plotted |
| `xlab` | a title for the x axis |
| `ylab` | a title for the y axis |
| `curve.add` | if `TRUE` a smooth curve is fit to the data and displayed. The function `loess` is used to fit the curve. |
| `curve.col` | a numeric value that determines the color of the smooth curve |
| `curve.span` | is passed on to `loess` as the `span` argument |
| `curve.lwd` | the line width for the smooth curve |
| `curve.n` | a numeric value that determines the sample size used to fit the curve. This makes fitting the curve faster with large datasets |
| `...` | further arguments passed to `plot` |

## Author(s)

Rafael A. Irizarry

## Examples

```
n <- 10000
signal <- runif(n,4,15)
bias <- (signal/5 - 2)^2
x <- signal + rnorm(n)
y <- signal + bias + rnorm(n)
maplot(x,y)
```

---

# `mypar` *mypar*

## Description

Called without arguments, this function optimizes graphical parameters for the RStudio plot window. `bigpar` uses big fonts which are good for presentations.

## Usage

```
mypar(a = 1, b = 1, brewer.n = 8, brewer.name = "Dark2", cex.lab = 1,
  cex.main = 1.2, cex.axis = 1, mar = c(2.5, 2.5, 1.6, 1.1),
  mgp = c(1.5, 0.5, 0), ...)
```

### Arguments

| | |
|---|---|
| `a` | the first entry of the vector passed to `mar` |
| `b` | the second entry of the vector passed to `mar` |
| `brewer.n` | parameter `n` passed to `brewer.pal` |
| `brewer.name` | parameters `name` passed to `brewer.pal` |
| `cex.lab` | passed on to `par` |
| `cex.main` | passed on to `par` |
| `cex.axis` | passed on to `par` |
| `mar` | passed on to `par` |
| `mgp` | passed on to `par` |
| `...` | other parameters passed on to `par` |

### Author(s)

Rafael A. Irizarry

### Examples

```
mypar()
plot(cars)
bigpar()
plot(cars)
```

---

## `myplclust` *plclust in colour*

### Description

Modifiction of plclust for plotting hclust objects in *in colour*!

### Usage

```
myplclust(hclust, labels = hclust$labels, lab.col = rep(1,
  length(hclust$labels)), hang = 0.1, xlab = "", sub = "", ...)
```

### Arguments

| | |
|---|---|
| `hclust` | hclust object |
| `labels` | a character vector of labels of the leaves of the tree |
| `lab.col` | colour for the labels; NA=default device foreground colour |
| `hang` | as in `hclust` & `plclust` |
| `xlab` | title for x-axis (defaults to no title) |
| `sub` | subtitle (defualts to no subtitle) |
| `...` | further arguments passed to `plot` |

### Author(s)

Eva KF Chan

---

## `nullplot` *nullplot*

### Description

Make an plot with nothing in it

### Usage

```
nullplot(x1 = 0, x2 = 1, y1 = 0, y2 = 1, xlab = "", ylab = "", ...)
```

### Arguments

| | |
|---|---|
| `x1` | lowest x-axis value |
| `x2` | largest x-axis value |
| `y1` | lowest y-axis value |
| `y2` | largest y-axis value |
| `xlab` | x-axis title, defaults to no title |
| `ylab` | y-axis title, defaults to no title |
| `...` | further arguments passed on to plot |

---

## `peek` *peek at the top of a text file*

### Description

this returns a character vector which shows the top n lines of a file

### Usage

```
peek(x, n = 2)
```

### Arguments

| | |
|---|---|
| `x` | a filename |
| `n` | the number of lines to return |

### Author(s)

Michael I. Love

## `popsd` *population standard deviation*

### Description

Returns the population variance. Note that `sd` returns the unbiased sample estimate of the population varaince. It simply multiplies the result of `var` by (n-1) / n with n the populaton size and takes the square root.

### Usage

```
popsd(x, na.rm = FALSE)
```

### Arguments

| | |
|---|---|
| `x` | a numeric vector or an R object which is coercible to one by `as.vector(x, "numeric")`. |
| `na.rm` | logical. Should missing values be removed? |

## `popvar` *population variance*

### Description

Returns the population variance. Note that `var` returns the unbiased sample estimate of the population varaince. It simply multiplies the result of `var` by (n-1) / n with n the populaton size.

### Usage

```
popvar(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | a numeric vector, matrix or data frame. |
| `...` | further arguments passed along to `var` |

---

## `sboxplot` *smart boxplot*

### Description

draws points or boxes depending on sample size

### Usage

```
sboxplot(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | a named list of numeric vectors |
| `...` | further arguments passed on to `boxplot` |

### Examples

```
sboxplot(list(a=rnorm(15),b=rnorm(75),c=rnorm(10000)))
```

---

## `shist` *smooth histogram*

### Description

a smooth histogram with unit indicator (we're simply scaling the kernel density estimate). The advantage of this plot is its interpretability since the height of the curve represents the frequency of a interval of size `unit` around the point in question. Another advantage is that if `z` is a matrix, curves are plotted together.

### Usage

```
shist(z, unit, bw = "nrd0", n, from, to, plotHist = FALSE, add = FALSE,
  xlab, ylab = "Frequency", xlim, ylim, main, ...)
```

### Arguments

| | |
|---|---|
| `z` | the data |
| `unit` | the unit which determines the y axis scaling and is drawn |
| `bw` | arguments to density |
| `n` | arguments to density |
| `from` | arguments to density |
| `to` | arguments to density |
| `plotHist` | a logical: should an actual histogram be drawn under curve? |

| | |
|---|---|
| `add` | a logical: add should the curve be added to existing plot? |
| `xlab` | x-axis title, defaults to no title |
| `ylab` | y-axis title, defaults to no title |
| `xlim` | range of the x-axis |
| `ylim` | range of the y-axis |
| `main` | an overall title for the plot: see `title`. |
| `...` | arguments to lines |

## Examples

```
set.seed(1)
x = rnorm(50)
par(mfrow=c(2,1))
hist(x, breaks=-5:5)
shist(x, unit=1, xlim=c(-5,5))
```

---

# `splitit` *split it*

## Description

Creates an list of indexes for each unique entry of `x`

## Usage

```
splitit(x)
```

## Arguments

| | |
|---|---|
| `x` | a vector |

## Examples

```
x <- c("a","a","b","a","b","c","b","b")
splitit(x)
```

---

## `splot` *smart plot*

---

### Description

if n > 10,000, make a random subset of 10,000 and plot. You can also specify a specific subset to plot. If length of subset is larger than `n`, a random sample is still used to reduce data size.

### Usage

```
splot(x, y, n = 10000, subset = NULL, xlab = NULL, ylab = NULL, ...)
```

### Arguments

| | |
|---|---|
| `x` | the x data |
| `y` | the y data |
| `n` | the number to subset |
| `subset` | explicit subset index (optional). |
| `xlab` | title for the x-axis |
| `ylab` | title for the y-axis |
| `...` | further parameters passed on to `plot` |

### Examples

```
x <- rnorm(1e5)
y <- rnorm(1e5)
splot(x,y,pch=16,col=rgb(0,0,0,.25))
```

# Index