

# Package ‘rafalib’

October 14, 2022

**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

**Repository** CRAN

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

## R topics documented:

as.fumeric . . . . .	2
bartab . . . . .	2
imagemat . . . . .	3
imagesort . . . . .	4
install_bioc . . . . .	4
largeobj . . . . .	5
maplot . . . . .	5
mypar . . . . .	6
myplclust . . . . .	7
nullplot . . . . .	8
peek . . . . .	8
popsd . . . . .	9
popvar . . . . .	9
sboxplot . . . . .	10
shist . . . . .	10
splitit . . . . .	11
splot . . . . .	12

<b>Index</b>	<b>13</b>
--------------	-----------

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 levels 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	<i>image with sorted rows</i>
-----------	-------------------------------

---

**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	<i>Install or update Bioconductor and CRAN packages</i>
--------------	---

---

**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 <code>biocLite</code>
-----	--

**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 environment and lists the `n` largest.

**Usage**

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

**Arguments**

<code>n</code>	the number of objects to return
<code>units</code>	units to display, see <code>?object.size</code>

**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 are 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 <code>length(x)</code> is larger than <code>n</code> , the <code>x</code> and <code>y</code> 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 <code>loess</code> 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 <code>loess</code> as the <code>span</code> 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 <code>plot</code>

**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 <a href="#">brewer.pal</a>
brewer.name	parameters name passed to <a href="#">brewer.pal</a>
cex.lab	passed on to <a href="#">par</a>
cex.main	passed on to <a href="#">par</a>
cex.axis	passed on to <a href="#">par</a>
mar	passed on to <a href="#">par</a>
mgp	passed on to <a href="#">par</a>
...	other parameters passed on to <a href="#">par</a>

**Author(s)**

Rafael A. Irizarry

**Examples**

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

---

myplclust

*plclust in colour*

---

**Description**

Modification 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 <a href="#">hclust</a> & <a href="#">plclust</a>
xlab	title for x-axis (defaults to no title)
sub	subtitle (defaults to no subtitle)
...	further arguments passed to <a href="#">plot</a>

**Author(s)**

Eva KF Chan

---

nullplot	<i>nullplot</i>
----------	-----------------

---

**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	<i>peek at the top of a text file</i>
------	---------------------------------------

---

**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	<i>population standard deviation</i>
-------	--------------------------------------

---

**Description**

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

**Usage**

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

**Arguments**

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

---

popvar	<i>population variance</i>
--------	----------------------------

---

**Description**

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

**Usage**

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

**Arguments**

<code>x</code>	a numeric vector, matrix or data frame.
<code>...</code>	further arguments passed along to <code>var</code>

---

sboxplot	<i>smart boxplot</i>
----------	----------------------

---

**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 <a href="#">boxplot</a>

**Examples**

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

---

shist	<i>smooth histogram</i>
-------	-------------------------

---

**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 <a href="#">title</a> .
...	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	<i>split it</i>
---------	-----------------

---

### 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**

<code>x</code>	the x data
<code>y</code>	the y data
<code>n</code>	the number to subset
<code>subset</code>	explicit subset index (optional).
<code>xlab</code>	title for the x-axis
<code>ylab</code>	title for the y-axis
<code>...</code>	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

as.fumeric, 2

barplot, 2  
bartab, 2  
bigpar (mypar), 6  
boxplot, 10  
brewer.pal, 7

hclust, 7

imagemat, 3  
imagesort, 4  
install\_bioc, 4

largeobj, 5  
loess, 6

maplot, 5  
mypar, 6  
myplclust, 7

nullplot, 8

par, 7  
peek, 8  
plclust, 7  
plot, 6, 7  
popsd, 9  
popvar, 9

sboxplot, 10  
sd, 9  
shist, 10  
splitit, 11  
splot, 12

title, 11

var, 9