

# Package ‘beginr’

October 12, 2022

**Version** 0.1.7

**Date** 2019-05-01

**Title** Functions for R Beginners

**Author** Peng Zhao

**Maintainer** Peng Zhao <pzhao@pzhao.net>

**Depends** R (>= 3.1.0)

**Imports** cranlogs (>= 2.1.0),

**Suggests**

**Description** Useful functions for R beginners, including hints for the arguments of the 'plot()' function, self-defined functions for error bars, user-customized pair plots and hist plots, enhanced linear regression figures, etc.. This package could be helpful to R experts as well.

**License** MIT + file LICENSE

**URL** <https://github.com/pzhaonet/beginr>

**BugReports** <https://github.com/pzhaonet/beginr/issues>

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-05-02 11:40:19 UTC

## R topics documented:

bib	2
dfplot	3
dfplot2	4
errorbar	5
list2ascii	6
lmdf	6
mf_skewness	7
name	7
packr	8

plotblank	9
plotcolorbar	9
plotcolors	10
plothist	10
plotlm	11
plotty	12
plotpairs	12
plotpairs2	13
plotpch	14
plotpkg	14
plottype	15
readdir	16
rpkg	16
rplc	17
se	17
tapply2	18
tapplydf	18
tapplydfv	19
writefile	19

## Index 20

---

bib	<i>Create a bib file for R packages, including the citations of user-defined packages.</i>
-----	--

---

### Description

Create a bib file for R packages, including the citations of user-defined packages.

### Usage

```
bib(pkg = c("base"), bibfile = "")
```

### Arguments

pkg	character. Packages
bibfile	character. File path and name to save the bib entries. If "" (the default), it prints to the standard output connection, the console unless redirected by sink.

### Value

bib entries

### Examples

```
bib()
bib(pkg = c("mindr", "bookdownplus", "pinyin"))
```

---

dfplot *Plot a dataframe, multiple ys against one x*

---

### Description

Plot a dataframe, multiple ys against one x

### Usage

```
dfplot(x, y, add = FALSE, xlab = "", ylab = "", myaxes = FALSE, xlim = NULL,
       ylim = NULL, mycol = NULL, mytype = "l", mypch = 20, mycex = 1, mylty = NULL,
       lwd = 1, xerror = NULL, yerror = NULL, mycolerrorbar = NULL, mylegend = NULL,
       mylegendcol = mycol, mylegendcex = 1, legendpos = "top")
```

### Arguments

x	a vector
y	a vector or a dataframe with the same length as x
add	logical, whether to add this plot to the previous one
xlab	character
ylab	character
myaxes	logical, whether to display axes automatically
xlim	numeric
ylim	numeric
mycol	colours
mytype	character
mypch	numeric or character
mycex	numeric
mylty	numeric
lwd	numeric
xerror	errorbar, same dimension of x
yerror	same dimension of y
mycolerrorbar	error bar colours
mylegend	character
mylegendcol	colors
mylegendcex	numeric
legendpos	character

### Value

a figure

**Examples**

```
x <- seq(0, 2 * pi, length.out = 100)
y <- data.frame(sin(x), cos(x))
yerror <- data.frame(abs(rnorm(100, sd = 0.3)), abs(rnorm(100, sd = 0.1)))
dfplot(x, y, yerror = yerror)
```

dfplot2

*Plot a dataframe, one y against multiple xs***Description**

Plot a dataframe, one y against multiple xs

**Usage**

```
dfplot2(x, y, xlab = "x", ylab = "y", xlim = NULL, ylim = NULL, mycol = NULL,
        mylty = NULL, xerror = NULL, yerror = NULL, mycolerrorbar = NULL, mylegend = NULL)
```

**Arguments**

x	a vector or a dataframe with the same length as x
y	a vector
xlab	character
ylab	character
xlim	numeric
ylim	numeric
mycol	colours
mylty	numeric
xerror	errorbar, same dimension of x
yerror	same dimension of y
mycolerrorbar	error bar colours
mylegend	character

**Value**

a figure

**Examples**

```
x <- seq(0, 2 * pi, length.out = 100)
y <- data.frame(sin(x), cos(x))
yerror <- data.frame(abs(rnorm(100, sd = 0.3)), abs(rnorm(100, sd = 0.1)))
dfplot2(y, x, xerror = yerror)
```

---

errorbar	<i>add error bars to a scatterplot.</i>
----------	---

---

**Description**

add error bars to a scatterplot.

**Usage**

```
errorbar(x, y, xupper = NULL, xlower = NULL, yupper = NULL, ylower = NULL,  
         col = "black", lty = 1)
```

**Arguments**

x	numeric
y	numeric
xupper	numeric
xlower	numeric
yupper	numeric
ylower	numeric
col	colors
lty	numeric

**Value**

errorbars in a plot

**Examples**

```
x <- seq(0, 2 * pi, length.out = 100)  
y <- sin(x)  
plot(x, y, type = "l")  
errorbar(x, y, yupper = 0.1, ylower = 0.1)
```

---

list2ascii	<i>Save a list into an ASCII file. in: a list. out: a file.</i>
------------	---

---

**Description**

Save a list into an ASCII file. in: a list. out: a file.

**Usage**

```
list2ascii(x, file = paste(deparse(substitute(x)), ".txt", sep = ""))
```

**Arguments**

x	a list
file	character. file name

**Value**

a file

**Examples**

```
alist <- list(a = 1:10, b = letters)
list2ascii(alist)
```

---

lmdf	<i>calculate linear regression between every two columns in a data frame. in: a dataframes. out: a dataframe showing the linear regression.</i>
------	---

---

**Description**

calculate linear regression between every two columns in a data frame. in: a dataframes. out: a dataframe showing the linear regression.

**Usage**

```
lmdf(data, simply = FALSE, intercept = TRUE)
```

**Arguments**

data	a dataframe
simply	logical
intercept	logical

**Value**

another dataframe

**Examples**

```
df <- data.frame(a = 1:10, b = 1:10 + rnorm(10), c = 1:10 + rnorm(10))
lmdf(df)
```

---

mf\_skewness

*Calculate the skewness of a distribution*

---

**Description**

Calculate the skewness of a distribution

**Usage**

```
mf_skewness(x)
```

**Arguments**

x                    the data to check

**Value**

the skewness of the distribution of x

**Examples**

```
mf_skewness(rnorm(100))
```

---

name

*Enhancement of names()*

---

**Description**

Enhancement of names()

**Usage**

```
name(data)
```

**Arguments**

data                a dataframe

**Value**

a list

**Examples**

```
df <- data.frame(a = NA, b = NA, c = NA)
name(df)
```

---

packr

*Create a package*

---

**Description**

Create a package

**Usage**

```
packr(pkg_name, packages, author = NULL, email = NULL, auto = FALSE, overwrite = FALSE)
```

**Arguments**

pkg_name	the name of the package which is to be created
packages	packages wrapped in this group
author	author of the new package
email	email of the author
auto	logical. whether to build and install the new package automatically
overwrite	logical. whether to overwrite the package with the same name if it already installed

**Value**

a folder with a package skeleton

**Examples**

```
## Not run:
packr("zhaor", c("mindr", "pinyin", "beginr", "bookdownplus", "steemr", "rmd"),
      "Your Name")

## End(Not run)
```

---

plotblank	<i>plot a blank figure</i>
-----------	----------------------------

---

**Description**

plot a blank figure

**Usage**

```
plotblank()
```

**Value**

a blank figure

**Examples**

```
plotblank()
```

---

plotcolorbar	<i>A reminder for color bars. More palettes can be found in 'colormap', 'RColorBrewer', and 'dichromat' packages.</i>
--------------	---

---

**Description**

A reminder for color bars. More palettes can be found in 'colormap', 'RColorBrewer', and 'dichromat' packages.

**Usage**

```
plotcolorbar()
```

**Value**

a figure

**Examples**

```
plotcolorbar()
```

---

plotcolors	<i>A reminder for colors</i>
------------	------------------------------

---

**Description**

A reminder for colors

**Usage**

```
plotcolors()
```

**Value**

a figure

**Examples**

```
plotcolors()
```

---

plothist	<i>Plot a user-customized hist</i>
----------	------------------------------------

---

**Description**

Plot a user-customized hist

**Usage**

```
plothist(data = rnorm(1000), mybreaks = "Sturges", myxlim = NULL, myylim = NULL,
  eightlines = TRUE, eightdigit = 0, eightcex = 0.8, eightcolors = c("red",
    "darkgreen", "blue", "black", "purple", "gold")[c(1, 2, 3, 2, 1, 6, 6,
      5, 4, 5)], mylegend = "", myxlab = "", return_df = FALSE, show_n = TRUE,
    show_skewness = TRUE, show_density = FALSE, show_normline = FALSE, x)
```

**Arguments**

data	a numeric vector
mybreaks	character
myxlim	numeric
myylim	numeric
eightlines	logical
eightdigit	numeric
eightcex	numeric

eightcolors	colors
mylegend	character
myxlab	character
return_df	logic
show_n	logical
show_skewness	logical
show_density	logical
show_normline	logical
x	a vector for plotting the curve

**Value**

a hist plot

**Examples**

```
plothist(rnorm(10000))
```

---

plotlm	<i>plot a linear regression figure and return a list of parameters.</i>
--------	---

---

**Description**

plot a linear regression figure and return a list of parameters.

**Usage**

```
plotlm(x, y, xlim = range(as.numeric(x), na.rm = TRUE), ylim = range(as.numeric(y),
  na.rm = TRUE), plot.title = "linear regression", xlab = "x", ylab = "y",
  refline = FALSE, slope = 1, intercept = 0, showr2 = TRUE, showleg = TRUE)
```

**Arguments**

x	numeric
y	numeric
xlim	numeric
ylim	numeric
plot.title	character
xlab	character
ylab	character
refline	logical. if a reference line is plotted
slope	slope of refline
intercept	intercept of refline
showr2	logical
showleg	logical

**Value**

a figure

**Examples**

```
plotlm(1:10, 1:10 + rnorm(10))
```

---

plotlty	<i>A reminder for lty</i>
---------	---------------------------

---

**Description**

A reminder for lty

**Usage**

```
plotlty(myld = 1)
```

**Arguments**

myld                numeric. line width

**Value**

a figure reminding you lty

**Examples**

```
plotlty()
```

---

plotpairs	<i>plot pair-wise correlations. in: a dataframe. out: a figure.</i>
-----------	---

---

**Description**

plot pair-wise correlations. in: a dataframe. out: a figure.

**Usage**

```
plotpairs(data, lower.panel = c(panel.lm, panel.smooth)[[1]], upper.panel = panel.cor,
  diag.panel = panel.diag, lwd = 2, col = "grey", labels = names(data), cex.labels = 4)
```

**Arguments**

data	a dataframe
lower.panel	can be panel.lm or panel.smooth
upper.panel	panel.cor
diag.panel	panel.diag
lwd	numeric
col	colors
labels	character
cex.labels	character

**Value**

a pair plot

**Examples**

```
df <- data.frame(a = 1:10, b = 1:10 + rnorm(10), c = 1:10 + rnorm(10))
plotpairs(df)
```

---

plotpairs2	<i>plot pair-wise correlations with p value. in: a dataframe. out: a figure.</i>
------------	--

---

**Description**

plot pair-wise correlations with p value. in: a dataframe. out: a figure.

**Usage**

```
plotpairs2(data, lower.panel = panel.smooth, upper.panel = panel.cor,
  diag.panel = panel.diag, lwd = 2, col = "grey", labels = "", cex.labels = 4)
```

**Arguments**

data	a dataframe
lower.panel	can be panel.lm or panel.smooth
upper.panel	panel.cor
diag.panel	panel.diag
lwd	numeric
col	colors
labels	character
cex.labels	character

**Value**

a pair plot

**Examples**

```
df <- data.frame(a = 1:10, b = 1:10 + rnorm(10), c = 1:10 + rnorm(10))
plotpairs2(df)
```

---

plotpch	<i>A reminder for pch</i>
---------	---------------------------

---

**Description**

A reminder for pch

**Usage**

```
plotpch(mycex = 5)
```

**Arguments**

mycex                    cex

**Value**

a figure reminding you pch

**Examples**

```
plotpch()
```

---

plotpkg	<i>plot daily download counts of packages</i>
---------	---

---

**Description**

plot daily download counts of packages

**Usage**

```
plotpkg(mypkg = "bookdownplus", from = Sys.Date() - 30, to = Sys.Date(), type = "o",
        pch = 19, col = "blue", cex = 1, textcex = 5)
```

**Arguments**

mypkg	character vector of package names.
from	character in 'Y-m-d'
to	character in 'Y-m-d'
type	the same as that in 'plot()'
pch	the same as that in 'plot()'
col	the same as that in 'plot()'
cex	the same as that in 'plot()'
textcex	cex of the package name

**Value**

a figure

**Examples**

```
plotpkg(mypkg = "rmarkdown")
```

---

plottype *A reminder for type*

---

**Description**

A reminder for type

**Usage**

```
plottype()
```

**Value**

a figure reminding you type

**Examples**

```
plottype()
```

---

readdir	<i>Read multiple tables into a list.</i>
---------	--

---

**Description**

Read multiple tables into a list.

**Usage**

```
readdir(mydir = getwd(), sep = c(", "), output = c("list", "data.frame"), header = TRUE,
        skip = 0)
```

**Arguments**

mydir	the folder path
sep	the field separator character.
output	the type of the output. 'list' or 'data.frame'.
header	logical. Indicating whether the file contains the names of the variables as its first line.
skip	the number of lines of the data file to skip before beginning to read data.

**Value**

a list or a data frame

---

rpkg	<i>Create a new R package demo folder</i>
------	---

---

**Description**

Create a new R package demo folder

**Usage**

```
rpkg()
```

**Value**

a folder with an R package skeleton

**Examples**

```
rpkg()
```

---

rplc	<i>Replace strings in a file</i>
------	----------------------------------

---

**Description**

Replace strings in a file

**Usage**

```
rplc(oldchar, newchar, filename)
```

**Arguments**

oldchar	old string
newchar	new string
filename	file name

**Value**

modified files

---

se	<i>standard error</i>
----	-----------------------

---

**Description**

standard error

**Usage**

```
se(x, na.rm = TRUE)
```

**Arguments**

x	numeric
na.rm	logical

**Value**

se

**Examples**

```
se(1:10)
```

---

tapply2	<i>a friendly version of tapply for a column in a dataframe</i>
---------	---

---

**Description**

a friendly version of tapply for a column in a dataframe

**Usage**

```
tapply2(data, select = names(data)[1], myfactor, ..., na.rm = c(TRUE, FALSE, NULL)[1])
```

**Arguments**

data	dataframe
select	character, column names to calc
myfactor	a colname as factor
...	function to apply to data
na.rm	logical

**Value**

a dataframe

---

tapplydf	<i>a friendly version of tapply for dataframes</i>
----------	--

---

**Description**

a friendly version of tapply for dataframes

**Usage**

```
tapplydf(data, select = names(data), myfactor, ..., na.rm = c(TRUE, FALSE, NULL)[1])
```

**Arguments**

data	dataframe
select	character, column names to calc
myfactor	a colname as factor
...	function to apply to data
na.rm	logical

**Value**

a dataframe

---

tapplydfv	<i>a friendly version of tapply</i>
-----------	-------------------------------------

---

**Description**

a friendly version of tapply

**Usage**

```
tapplydfv(colname = "tapply", x, factor, ...)
```

**Arguments**

colname	character
x	a dataframe
factor	factor for tapply
...	the function to apply to data

**Value**

a dataframe

---

writefile	<i>save csv file with asking if the file already exists.</i>
-----------	--

---

**Description**

save csv file with asking if the file already exists.

**Usage**

```
writefile(data, writefile, row.names = FALSE)
```

**Arguments**

data	a data frame
writefile	destination file
row.names	logical

**Value**

write a file

# Index

`bib`, 2

`dfplot`, 3

`dfplot2`, 4

`errorbar`, 5

`list2ascii`, 6

`lmdf`, 6

`mf_skewness`, 7

`name`, 7

`packr`, 8

`plotblank`, 9

`plotcolorbar`, 9

`plotcolors`, 10

`plothist`, 10

`plotlm`, 11

`plotlty`, 12

`plotpairs`, 12

`plotpairs2`, 13

`plotpch`, 14

`plotpkg`, 14

`plotype`, 15

`readdir`, 16

`rpkg`, 16

`rplc`, 17

`se`, 17

`tapply2`, 18

`tapplydf`, 18

`tapplydfv`, 19

`writefile`, 19