Package 'Jmisc'

July 21, 2025

Type Package

Title Julian Miscellaneous Function

Version 0.3.1.1

Date 2011-12-26

Author TszKin Julian Chan <ctszkin@gmail.com>

Maintainer TszKin Julian Chan <ctszkin@gmail.com>

Description Some handy function in R.

License GPL (>= 2)

LazyLoad yes

Suggests testthat

Contact TszKin Julian Chan <ctszkin@gmail.com>

NeedsCompilation no

Repository CRAN

Date/Publication 2022-06-22 05:53:25 UTC

Contents

---

addCol *Add a constant column to a data.frame or matrix*

Description

Add a constant column to data.frame or matrix.

Usage

```
addCol(x, ..., value)
```

Arguments

x	data.frame or matrix
...	constants
value	vector a vector of constants

Value

a data.frame or matrix contains all columns in x and those constant columns.

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
d=data.frame(x=1:5,y=11:15)
addCol(d,a=1,b=2,c=3)
addCol(d,value=c(a=100,b=200,c=300))
```

---

demean *Demean a vector or a matrix (by column)*

Description

Demean a vector or a matrix (by column)

Usage

```
demean(x)
```

Arguments

x	Vector or matrix

Value

Demeaned value of x

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
x<-matrix(1:20,ncol=2)
demean(x)
```

`evalFunctionOnList` *Evaluate Function Under Local Variables*

Description

This function evaluates a function x under an environment which is created by a list. All elements of the list is local to the function; other words all elements of the list can be accessed directly by the function. A new environment is created and each element of `variables` is assigned to the new environment. Then the environment associated with the x is updated with the new environment. Finally `x(...)` is evaluated and return the result.

Usage

```
evalFunctionOnList(x, variables = list(), ..., parent_env)
```

Arguments

x	A function to be called
variables	A list to be converted to an environment
...	Further arguments to x
parent_env	parent environment

Value

Return value of the `x(...)`.

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

environment

Examples

```
evalFunctionOnList(function() rnorm(n,mean,sd),list(n=5,mean=5,sd=1))
```

generateSignificance *Generate t-statistics, p-value and significance*

Description

Generate t-statistics, p-value and significance from estimates and its sd. Estimates and its SD is the first and second column respectively

Usage

```
generateSignificance(x, row_names)
```

Arguments

x	A matrix or data.frame
row_names	names of row

Value

a data.frame

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
n<-1000
x_data<-cbind(rnorm(n,mean=0),rnorm(n,mean=1))
x_estimates<-cbind(apply(x_data,2,mean),apply(x_data,2,sd)/sqrt(n))
generateSignificance(x_estimates)
generateSignificance(x_estimates,row_names=c("mean0","mean1") )
```

JBTest *p Value of Jarque Bera test*

Description

Return the p Value of Jarque Bera test. The Jarque Bera test test the null hypothesis that the data are from a normal distribution.

Usage

```
JBTest(x)
```

Arguments

x	data

Value

p Value of Jarque Bera test

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
JBTest(rnorm(50))
JBTest(rt(50,3))

n=100
# size
mean(replicate(n,JBTest(rnorm(100)))<0.05)

# power
mean(replicate(n,JBTest(rt(100,3)))<0.05)
```

---

label_both_parsed_recode

Combine label_both and label_parsed in **ggplot2**.

Description

Combine label_both and label_parsed in **ggplot2**. Also added a rename function to it see label_both and label_parsed in **ggplot2** for details.

Usage

```
label_both_parsed_recode(display_name)
```

Arguments

display_name	A vector contains the display name. Names of the vector are the original name.

Value

A function similar to label_both and label_parsed in **ggplot2** for details.

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

References

https://CRAN.R-project.org/package=ggplot2

`oapply` *Outer apply*

Description

Outer apply It use the expand.grid to compute all possible combination of `X` and `Y`, then call the mapply with the combination generated and `FUN`.

Usage

```
oapply(X, Y, FUN, switch_order = FALSE, ...)
```

Arguments

`X`	first argument to `FUN`
`Y`	second argument to `FUN`
`FUN`	a function to apply. See mapply
`switch_order`	Switch the order of `X` and `Y` in expand.grid
`...`	other arguments to mapply

Value

same as mapply.

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

mapply

Examples

```
oapply(11:15,1:5,choose)
oapply(11:15,1:5,choose,switch_order=TRUE)
```

packages *load packages with auto-installation*

Description

load add-on packages. If the packages can not be found, install.packages is called.

Usage

```
packages(x, ...)
```

Arguments

x	name of the packages
...	arguments to install.packages

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

require install.packages

Examples

```
## Not run:
packages("foreach")

## End(Not run)
```

recode *Recode the value of a vector*

Description

Recode the value of a vector or matrix.

Usage

```
recode(x, from, to)
```

Arguments

x	a vector or matrix
from	original value of x
to	new value of x

Value

recoded x

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
x=rep(1:5,each=2)
recode(x,from=1:5,to=5:1)
recode(x,from=1:5,to=11:15)
```

---

repCol *Repeat a vector by col*

Description

Repeat a vector by col

Usage

```
repCol(x, n)
```

Arguments

x	vector or matrix
n	number of replication

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

repRow

Examples

```
repRow(c(a=1,b=2,c=3),5)
repCol(c(a=1,b=2,c=3),5)
```

repRow *Repeat a vector by row*

Description

Repeat a vector by row

Usage

```
repRow(x, n)
```

Arguments

x	vector or matrix
n	number of replication

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

repCol

Examples

```
repRow(c(a=1,b=2,c=3),5)
repCol(c(a=1,b=2,c=3),5)
```

shift *shift a vector by* shift_by *unit*

Description

Repeat a vector by row

Usage

```
shift(x, shift_by)
```

Arguments

x	a vector
shift_by	number of shift

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
d<-data.frame(x=1:15)
#generate lead variable
d$df_lead2<-shift(d$x,2)
#generate lag variable
d$df_lag2<-shift(d$x,-2)
```

---

sourceAll *Source all the R files of a directory*

Description

Source all file with extension .r or .R

Usage

```
sourceAll(path = ".", ...)
```

Arguments

path	path of the directory
...	other arguments to source

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

See Also

repCol

Examples

```
## Not run:
sourceAll()

## End(Not run)
```

splitBy *Split a vector by a sequence of length*

Description

Split a vector by a sequence of length This function will split the vector x into `length(x)` subvector. The length of each subvector is given by by.

Usage

```
splitBy(x, by)
```

Arguments

x	A vector to be splitted
by	A vector of length

Value

a list of subvector

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
splitBy((1:10)*10,c(2,2))
splitBy((1:10)*10,c(2,3,4))
## Not run:
expect_equivalent(splitBy((1:10)*10,c(2,2)) ,  list(c(10,20),c(30,40)))
expect_equivalent(splitBy((1:10)*10,c(2,3,4)) , list( c(10,20), c(30,40,50) ,c(60,70,80,90) ))

## End(Not run)
```

---

tic *Start Stop clock to measure performance*

Description

Start/clock to measure performance. Same as tic and toc in matlab

Usage

```
tic(name = ".time_Jmisc", envir = .GlobalEnv)

toc()
```

Arguments

name	Name of the temporary time variable
envir	environment of the temporary time variable

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
## Not run:
tic()
Sys.sleep(1)
toc

## End(Not run)
```

%+% *Concatenate two strings*

Description

Paste two strings together without separation.

Usage

```
s1 %+% s2
```

Arguments

s1	First String
s2	Second String

Value

```
paste(s1,s2,sep="")
```

Author(s)

TszKin Julian Chan <ctszkin@gmail.com>

Examples

```
cat("Hello" %+% "World")
```

Index