Package 'metools'

June 26, 2020

Title Macroeconomics Tools

Version 1.0.0

Description Provides a number of functions to facilitate the handling and production of reports using time series data.
The package was developed to be understandable for beginners, so some functions aim to transform processes that would be
complex into functions with a few lines. The main advantage of using the 'metools' package is the ease of producing reports and
working with time series using a few lines of code, so the code is clean and easy to understand/maintain.
Learn more about the 'metools' at <https://metoolsr.wordpress.com>.

License GPL-3

Encoding UTF-8

LazyData true

Imports ggplot2, scales, stringr, tibble, lubridate, tidyr

RoxygenNote 7.1.0

URL https://metoolsr.wordpress.com,https://github.com/jvg0mes/metools,https://jvg0mes.github.io/metoolsr

NeedsCompilation no

Author João Victor Gomes de Araujo Santana [aut, cre]

Maintainer João Victor Gomes de Araujo Santana <jvg.santana@gmail.com>

Repository CRAN

Date/Publication 2020-06-26 08:50:14 UTC

R topics documented:

`col2char` *Transform defined columns to character.*

Description

col2char transform columns type to character.

Usage

```
col2char(x, start, end = ncol(x))
```

Arguments

x	a dataframe
start	number of start column
end	number of last column (default=last)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c(3,2,5,6,5,4))
class(v[,1]) #here class is numeric
v=col2char(v,1)
class(v[,1]) #now class is character
```

---

col2factor *Transform defined columns to factor.*

Description

col2factor transform columns type to factor.

Usage

```
col2factor(x, start, end = ncol(x))
```

Arguments

x	a dataframe
start	number of start column
end	number of last column (default=last)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c(3,2,5,6,5,4))
class(v[,1]) #here class is numeric
v=col2factor(v,1)
class(v[,1]) #now class is character
```

col2num *Transform defined columns to numeric.*

Description

col2num transform columns type to numeric.

Usage

```
col2num(x, start, end = ncol(x))
```

Arguments

x	a dataframe
start	number of start column
end	number of last column (default=last)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c('3','2','5','6','5','4'))
class(v[,1]) #here class is factor
v=col2num(v,1)
class(v[,1]) #now class is character
```

col2percent *Add percent in column*

Description

col2percent transform columns to percent.

Usage

```
col2percent(x, start, end = ncol(x), mult100 = FALSE)
```

Arguments

x	a dataframe
start	number of start column
end	number of last column (default=last)
mult100	multiply by 100 if the number is a decimal fraction(T or F)(default=F)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c(15,5,20,50,10))
col2percent(v,start=1)

v=data.frame(c(0.15,0.05,0.2,0.5,0.1))
col2percent(v,start=1,mult100=TRUE)
```

`colpct2num` *Remove percent from a column, and transform in number*

Description

When use col2percent function to add a percent in a column, the type of this column now is character, colpct2num function remove percent from this column and transform in number.

Usage

```
colpct2num(x, start, end = ncol(x), div100 = TRUE)
```

Arguments

`x`	a dataframe
`start`	number of start column
`end`	number of last column (default=last)
`div100`	division by 100 (T or F)(default=T)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c(15,5,20,50,10))
v=col2percent(v,start=1)
v=colpct2num(v,start=1,div100=TRUE)
```

colround *Round defined columns*

Description

colround round defined columns.

Usage

```
colround(x, start, end = ncol(x), digits)
```

Arguments

x	a dataframe
start	number of start column
end	number of last column (default=last)
digits	number of round digits

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame(c(3.255,5.826,4.567,2.462))
v=colround(v,1,digits=1)
```

cuminyear *Accumulated variation in year*

Description

cuminyear calculates an accumulated variation in year of a index. Data must be start in january, use start to set this, if you data don't start in january and you need use this values, consider complete the previous months with 0.

Usage

```
cuminyear(data, coldate, colnum, start = 1)
```

Arguments

data	a dataframe
coldate	number of date column
colnum	number of values column
start	number of start row

Value

Return a dataframe.

Examples

```
v=data.frame(
"Date"=c(seq.Date(as.Date("2018-01-01"),as.Date("2019-12-01"),by='month'))
,"Value"=c(rep(2,6),rep(3,6),rep(1,6),rep(5,6)))
cuminyear(v,coldate=1,colnum=2)

v=data.frame(
"Date"=c(seq.Date(as.Date("2018-06-01"),as.Date("2019-12-01"),by='month'))
,"Value"=c(rep(3,7),rep(1,6),rep(5,6)))

#this case, we can start in january 2019
cuminyear(v,coldate=1,colnum=2,start=8)

#or if we need the previous values i can complete january 2018 to may 2018 with 0.
v1=data.frame(Date=c(seq.Date(as.Date("2018-01-01"),as.Date("2018-05-01"),by='month')),
"Value"=c(rep(0,5)))
v=rbind(v1,v)
cuminyear(v,coldate=1,colnum=2)
```

---

cuminyear_var *Accumulated variation in year*

Description

cuminyear_var calculates an accumulated variation in year of a rate, _var means the data must be a percentage variation. Data must be start in january, if you data don't start in january and you need use this values, consider complete the previous months with 0.

Usage

```
cuminyear_var(data, coldate, colnum, div100 = FALSE)
```

Arguments

data	a dataframe
coldate	number of date column
colnum	number of values column
div100	divide data by 100, use if data is not fraction

Value

Return a dataframe.

Examples

```
v=data.frame(
"Date"=c(seq.Date(as.Date("2018-01-01"),as.Date("2019-12-01"),by='month'))
,"Value"=c(rep(0.02,12),rep(0.03,12)))
cuminyear_var(v,coldate=1,colnum=2)

v=data.frame(
"Date"=c('january','february','march')
,"Value"=c('1%','3%','2%'))
v=colpct2num(v,start=2,div100=TRUE)
v[[1]]=month2num(v[[1]])
v[[1]]=paste('2018',v[[1]],'01',sep="-")
v[[1]]=as.Date(v[[1]])
cuminyear_var(v,coldate=1,colnum=2)
```

---

cum_var *Accumulated variation*

Description

cum_var calculates an accumulated variation of a rate, _var means the data must be a percentage variation.

Usage

```
cum_var(data, colnum, t, div100 = FALSE)
```

Arguments

data	a dataframe
colnum	number of column
t	number of periods to accumulate
div100	divide data by 100, use if data is not fraction

Value

Return a dataframe.

Examples

```
v=data.frame(c(0.03,0.02,0.05))
cum_var(v,colnum=1,t=3)

v=data.frame(c('3%','2%','5%'))
v=colpct2num(v,start=1,div100=TRUE)
cum_var(v,colnum=1,t=3)
```

`gm.col` *Bar Graphic Model*

Description

gm.col make a bar plot. Graphic models function family do graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.col_ord but the graphic customize is more limited.

Usage

```
gm.col(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  xlab = NULL,
  ylab = NULL,
  div100 = FALSE,
  percent = FALSE,
  fontsize = 0,
  cserie = "#17B221",
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground
)
```

Arguments

`data`	a dataframe
`ncolx`	number of x column in data frame
`ncoly`	number of y column in data frame

ntimes	number of observations to plot (count by tail)
title	title of plot
xlab	x axis label
ylab	y axis label
div100	If data percent are not in decimal format set TRUE.
percent	If TRUE, y axis in percent (default=F)
fontsize	change size of all words in graphic (only numbers)
cserie	change color of serie
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq(from=1,to=4,by=1),"y"=c(5,3,7,2))

gm.col(v,1,2,title="Simple example",ntimes=3)
```

`gm.col_ord` *Ordered Bar Graphic Model*

Description

gm.col_ord make a ordered bar plot. Graphic models function family do graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.col_ord but the graphic customize is more limited.

Usage

```
gm.col_ord(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  xlab = NULL,
  ylab = NULL,
  percent = FALSE,
```

```
  div100 = FALSE,
  dec = FALSE,
  fontsize = 0,
  cserie = "#17B221",
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground
)
```

Arguments

data	a dataframe
ncolx	number of x column in data frame
ncoly	number of y column in data frame
ntimes	number of observations to plot (count by tail)
title	title of plot
xlab	x axis label
ylab	y axis label
percent	If TRUE, y axis in percent (default=F)
div100	If data percent are not in decimal format set TRUE.
dec	If TRUE, bars plot in decrescent order.
fontsize	change size of all words in graphic (only numbers)
cserie	change color of serie
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq(from=1,to=4,by=1),"y"=c(5,3,7,2))

gm.col_ord(v,1,2,title="Simple example",ntimes=3)
```

gm.col_ord_wl *Ordered Bar Graphic with Legend Model*

Description

gm.col_ord_wl make a bar plot. Graphic models function family do graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.col_ord but the graphic customize is more limited.

Usage

```
gm.col_ord_wl(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  legtitle,
  xlab = NULL,
  ylab = NULL,
  dec = FALSE,
  div100 = FALSE,
  percent = FALSE,
  fontsize = 0,
  colors = grDevices::rainbow(n = ntimes, v = 0.7),
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground,
  legwpos = 0,
  legheight = 0.5
)
```

Arguments

data	a dataframe
ncolx	number of x column in data frame
ncoly	number of y column in data frame
ntimes	number of observations to plot (count by tail)
title	title of plot
legtitle	title of legendbox
xlab	x axis label
ylab	y axis label
dec	If TRUE serie come be decrescent,if FALSE crescent(default=F)

div100	If data percent are not in decimal format set TRUE.
percent	If TRUE, y axis in percent (default=F)
fontsize	change size of all words in graphic (only numbers)
colors	colors of bars
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)
legwpos	legend words position (numeric)
legheight	height of legend box

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq(from=1,to=4,by=1),"y"=c(5,3,7,2))

gm.col_ord_wl(v,1,2,title="Simple example",ntimes=3,legwpos=-2.5)
```

---

gm.col_wl *Bar Graphic with Legend Model*

Description

gm.col_wl make a bar plot. Graphic models function family do graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.col_ord but the graphic customize is more limited.

Usage

```
gm.col_wl(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  legtitle,
  xlab = NULL,
  ylab = NULL,
  div100 = FALSE,
  percent = FALSE,
  fontsize = 0,
```

```
  colors = grDevices::rainbow(n = ntimes, v = 0.7),
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground,
  legwpos = 0,
  legheight = 0.5
)
```

Arguments

data	a dataframe
ncolx	number of x column in data frame
ncoly	number of y column in data frame
ntimes	number of observations to plot (count by tail)
title	title of plot
legtitle	title of legendbox
xlab	x axis label
ylab	y axis label
div100	If data percent are not in decimal format set TRUE.
percent	If TRUE, y axis in percent (default=F)
fontsize	change size of all words in graphic (only numbers)
colors	colors of bars
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)
legwpos	legend words position (numeric)
legheight	height of legend box

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq(from=1,to=4,by=1),"y"=c(5,3,7,2))

gm.col_wl(v,1,2,title="Simple example",ntimes=3,legwpos=-2.5)
```

`gm.line` *Line Graphic Model*

Description

gm.line make a line plot. Graphic models function family do graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.line but the graphic customize is more limited.

Usage

```
gm.line(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  xlab = NULL,
  ylab = NULL,
  div100 = FALSE,
  percent = FALSE,
  fontsize = 0,
  lwdserie = 1.5,
  cserie = "white",
  clines = "white",
  ctext = "white",
  cbackground = "#141414"
)
```

Arguments

`data`	a dataframe
`ncolx`	number of x column in data frame
`ncoly`	number of y column in data frame
`ntimes`	number of observations to plot (count by tail)
`title`	title of plot
`xlab`	x axis label
`ylab`	y axis label
`div100`	If data percent are not in decimal format set TRUE.
`percent`	If TRUE, y axis in percent (default=F)
`fontsize`	change size of all words in graphic (only numbers)
`lwdserie`	size of serie
`cserie`	change color of serie

clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq(from=1,to=4,by=1),"y"=c(5,3,7,2))

gm.line(v,1,2,title="Simple example",ntimes=3)
```

---

gm.tscol *Time serie bar Graphic Model*

Description

gm.tscol make a bar plot in time serie format. Graphic models function family make graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.tscol but the graphic customize is more limited . The data don't need be a ts object.

Usage

```
gm.tscol(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  ylab = NULL,
  percent = FALSE,
  div100 = FALSE,
  fontsize = 0,
  datebreaks = "1 months",
  dateformat = "%b/%y",
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground
)
```

Arguments

data	a dataframe
ncolx	number of x column in data frame
ncoly	number of y column in data frame
ntimes	number of observations to plot (count by tail)
title	title of plot
ylab	y axis label
percent	If TRUE y axis in percent (default=F)
div100	If data percent are not in decimal format set TRUE.
fontsize	change size of all words in graphic (only numbers)
datebreaks	datebreaks in x axis (default="1 month")
dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(5,3,7,2))

gm.tscol(v,1,2,title="Simple example",ntimes=3)
```

---

gm.tscol2 *Time serie bar Graphic Model*

Description

gm.tscol2 make a bar plot in time serie format. The difference between gm.tscol2 and gm.tscol is possibility to select serie color. Graphic models function family make graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.tscol but the graphic customize is more limited. The data don't need be a ts object.

Usage

```
gm.tscol2(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  ylab = NULL,
  percent = FALSE,
  div100 = FALSE,
  fontsize = 0,
  datebreaks = "1 months",
  dateformat = "%b/%y",
  cserie = "white",
  clines = "white",
  ctext = "white",
  cbackground = "#141414",
  cbserie = cbackground
)
```

Arguments

data	a dataframe
ncolx	number of x column in data frame
ncoly	number of y column in data frame
ntimes	number of observations to plot (count by tail)
title	title of plot
ylab	y axis label
percent	If TRUE y axis in percent (default=F)
div100	If data percent are not in decimal format set TRUE.
fontsize	change size of all words in graphic (only numbers)
datebreaks	datebreaks in x axis (default="1 month")
dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
cserie	color of serie
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background
cbserie	color of serie border (default= same cbackground)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(5,3,7,2))

gm.tscol2(v,1,2,title="Simple example",ntimes=3)
```

`gm.tsl` *Time serie line Graphic Model*

Description

gm.tsl make a line plot in time serie format. Graphic models function family make graphic creation easy, is recommended for new programers, they have less and easyful parameters then p.tsl but the graphic customize is more limited. The data don't need be a ts object.

Usage

```
gm.tsl(
  data,
  ncolx,
  ncoly,
  ntimes,
  title,
  ylab = NULL,
  percent = FALSE,
  div100 = FALSE,
  fontsize = 0,
  lwdserie = 1,
  datebreaks = "1 months",
  dateformat = "%b/%y",
  cserie = "white",
  clines = "white",
  ctext = "white",
  cbackground = "#141414"
)
```

Arguments

`data`	a dataframe
`ncolx`	number of x column in data frame
`ncoly`	number of y column in data frame
`ntimes`	number of observations to plot (count by tail)
`title`	title of plot
`ylab`	y axis label

percent	If TRUE y axis in percent (default=F)
div100	If data percent are not in decimal format set TRUE.
fontsize	change size of all words in graphic (only numbers)
lwdserie	size of serie
datebreaks	datebreaks in x axis (default="1 month")
dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
cserie	color of serie
clines	color of lines in graphic
ctext	color of words in graphic
cbackground	color of graphic background

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(5,3,7,2))

gm.tsl(v,1,2,title="Simple example",ntimes=3)
```

---

me.lag *Lag a data*

Description

me.lag lag a vector if t>0 or lead a vector if t<0.

Usage

```
me.lag(x, t = 1, nafill = NA, extrapolate = FALSE)
```

Arguments

x	a vector
t	number of times to lag (default=1)
nafill	set value to fill NA's before first t value
extrapolate	if TRUE extrapolate excedent values, only if t>0 (default=FALSE)

Value

Return a vector.

Examples

```
v=c(3,2,5,6,5,4)
me.lag(v)

#now lead

me.lag(v,t=-1)
```

`me.spread` *Spread an dataframe.*

Description

Transforms columns into rows and rows into columns.

Usage

```
me.spread(data, namenc = " ", mode = FALSE)
```

Arguments

`data`	a dataframe
`namenc`	name of new column (first column) (default="")
`mode`	if results are incorretly try set this to TRUE

Value

Return a dataframe.

Examples

```
v=data.frame('date'=c('2016','2017'),'value1'=c(12,10),'value2'=c(8,6))
me.spread(v,namenc='old header')
```

metools *Macroeconomics Tools*

Description

The 'metools' package provides a number of functions to facilitate the handling and production of reports using time series data. The package was developed to be understandable for beginners, so some functions aim to transform processes that would be complex into functions with a few lines. The main advantage of using the 'metools' package is the ease of producing reports and working with time series using a few lines of code, so the code is clean and easy to understand/maintain. Learn more about the 'metools' at <https://metoolsr.wordpress.com>.

Details

metools: A package for work with macroeconomics time series.

The 'metools' package provide two categorys of functions: Data manipulate: don't have prefix. Graphics: have p. prefix.

Author(s)

João Victor Gomes (jvg.santana@gmail.com)

See Also

Useful links:

https://metoolsr.wordpress.com https://github.com/jvg0mes/metools https://jvg0mes.github.io/metoolsr

metools.help *Metools Help*

Description

Use this function to receive help to use metools.

Usage

```
metools.help()
```

Value

Return a info.

Examples

```
metools.help()
```

month2num *Transform month names to month numbers*

Description

month2num transform month names to month numbers

Usage

```
month2num(date)
```

Arguments

date	a month names vector

Value

Return a month numbers.

Examples

```
v=c("jan","fev","mar","abr","mai","jun","jul","ago","set","out","nov","dez")
month2num(v)

v=data.frame('date'=c("janeiro","fevereiro",'marÃ§o','abril'),'values'=c(18,27,10,48))
month2num(v$date)
#or
month2num(v[[1]])

#you can substitute column with function:
v$date = month2num(v$date)
v[[1]] = month2num(v[[1]])
```

mp.s *Multi serie plot*

Description

mp.s make a plot with one or more series. The object parameter require a ggplot object (Look at the examples).

Usage

```
mp.s(
  object,
  xaxis,
  yaxis,
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cticks = "black",
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

object	a ggplot graphic object
xaxis	x axis of one of your graphics
yaxis	y axis of one of your graphics
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)

cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cticks	color of axis ticks
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a graphic.

Examples

```
v=data.frame("x"=c('a','b','c','d','e'),"y"=c(5,3,7,10,9),"y2"=c(7,2,5,8,7))

g= ggplot2::ggplot()+ggplot2::geom_line(mapping=ggplot2::aes(x=v$x,y=v$y,group=1),lwd=2)+
ggplot2::geom_line(mapping=ggplot2::aes(x=v$x,y=v$y2,group=1),color='blue',lwd=2)

mp.s(object=g,xaxis=v$x,yaxis=v$y,title="Simple example")

mp.s(g,v$x,v$y,percent=TRUE,title="Example with percent data",xlab=NULL,ylab=NULL)

mp.s(g,v$x,v$y,percent=TRUE,yaccuracy=1,title="y accuracy set",xlab=NULL,ylab=NULL)

g= ggplot2::ggplot()+ggplot2::geom_area(mapping=ggplot2::aes(x=v$x,y=v$y),
fill='red',lwd=2,group=1)+
ggplot2::geom_area(mapping=ggplot2::aes(x=v$x,y=v$y2),fill='blue',lwd=2,group=1)

mp.s(g,v$x,v$y,title="Example with area plot")

v=data.frame("x"=c('a','b','c','d','e'),"y"=c(5,-3,-6,10,7))

g= ggplot2::ggplot()+ggplot2::geom_col(ggplot2::aes(x=v$x,y=v$y,group=1),
fill=p.colorbypositive(v$y),color='black',lwd=1)+
ggplot2::geom_line(ggplot2::aes(x=v$x,y=v$y,group=1),color='black',lwd=1)

mp.s(g,v$x,v$y,title="Example with colorbypositive",xlab=NULL,ylab=NULL)
```

mp.ts *Multi serie plot in time serie format*

Description

mp.ts make plot in time serie format with one or more series. The data don't need be a ts object. The object parameter require a ggplot object (Look at the examples).

Usage

```
mp.ts(
  object,
  xaxis,
  yaxis,
  dateformat = "%Y-%m",
  datebreaks = "1 month",
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cticks = "black",
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

object	a ggplot graphic object
xaxis	x axis of one of your graphics
yaxis	y axis of one of your graphics
dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
datebreaks	datebreaks in x axis (default="1 month")

ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cticks	color of axis ticks
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-05-01'),by='month'),"y"=c(5,3,7,10,9),"y2"=c(7,2,5,8,7))

g= ggplot2::ggplot()+ggplot2::geom_line(mapping=ggplot2::aes(x=v$x,y=v$y),lwd=2)+
ggplot2::geom_line(mapping=ggplot2::aes(x=v$x,y=v$y2),color='blue',lwd=2)

mp.ts(object=g,xaxis=v$x,yaxis=v$y,title="Simple example")

mp.ts(g,v$x,v$y,percent=TRUE,title="Example with percent data",xlab=NULL,ylab=NULL)

mp.ts(g,v$x,v$y,percent=TRUE,yaccuracy=1,title="y accuracy set",xlab=NULL,ylab=NULL)

g= ggplot2::ggplot()+ggplot2::geom_area(mapping=ggplot2::aes(x=v$x,y=v$y),
fill='red',lwd=2)+
ggplot2::geom_area(mapping=ggplot2::aes(x=v$x,y=v$y2),fill='blue',lwd=2)
```

```
mp.ts(g,v$x,v$y,dateformat="%B",title="Example with area plot")

v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-05-01'),by='month'),"y"=c(5,-3,-6,10,7))

g= ggplot2::ggplot()+ggplot2::geom_col(ggplot2::aes(x=v$x,y=v$y),
fill=p.colorbypositive(v$y),color='black',lwd=1)+
ggplot2::geom_line(ggplot2::aes(x=v$x,y=v$y),color='black',lwd=1)

mp.ts(g,v$x,v$y,title="Example with colorbypositive",xlab=NULL,ylab=NULL)
```

---

num2month *Transform month numbers to month names*

Description

num2month transform month numbers to month names

Usage

```
num2month(date, abbreviate = FALSE, ptbr = FALSE)
```

Arguments

date	a month numbers vector
abbreviate	abbreviate months name
ptbr	transalate result to "Portugues (Brasil)".

Value

Return a month names.

Examples

```
v=c(01,02,03,04,05,06,07,08,09,10,11,12)
num2month(v)
num2month(v,abbreviate=TRUE)
num2month(v,abbreviate=FALSE,ptbr=TRUE)
num2month(v,abbreviate=TRUE,ptbr=TRUE)

v=data.frame('date'=c(01,02,03,04),'values'=c(18,27,10,48))
num2month(v$date)
#or
num2month(v[[1]])

#you can substitute column with function:
v$date = num2month(v$date)
v[[1]] = num2month(v[[1]])
```

```
#The data can be a string, but is recommended use numbers,
#see a string examples:
v=c('01','02','03','04','05','06','07','08','09','10','11','12')
num2month(v)

v=c('1','2','3','4','5','6','7','8','9','10','11','12')
num2month(v)
```

---

p.col *Bar plot*

Description

p.col make a bar plot.

Usage

```
p.col(
  data,
  xaxis,
  yaxis,
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cserie = "black",
  cbserie = cserie,
  cticks = "black",
  lwdserie = 1,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
```

```
  xlim = NULL
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cserie	color of serie
cbserie	color of serie border (default= same cserie)
cticks	color of axis ticks
lwdserie	size of serie
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a graphic.

Examples

```
v=data.frame("x"=1:5,"y"=c(10,4,8,5,2))
p.col(v,xaxis= v$x,yaxis=v$y)
#or
p.col(v,xaxis= v[[1]],yaxis=v[[2]])
```

p.colorbypositive *Color by positive or negative*

Description

p.colorbypositive is a function to create a vector with colors by positive or negative. Recommended to color graphics created with metools p.functions.

Usage

```
p.colorbypositive(x, colorp = "#17B221", colorn = "#B21717")
```

Arguments

x	a numeric vector
colorp	Positive values color (default=Green)
colorn	Negative values color (default=Red)

Value

Return a vector with colors.

Examples

```
v=c(-3,-2,2,-2,3,2)
p.colorbypositive(x=v,colorp="blue",colorn="grey")

barplot(v,col=p.colorbypositive(v))
```

p.colorbyvar *Color by variation*

Description

p.colorbyvar is a function to create a vector with colors by variation. Recommended to color graphics created with metools p.functions.

Usage

```
p.colorbyvar(x, colorp = "#17B221", colorn = "#B21717", lag = 1)
```

Arguments

x	a numeric vector
colorp	Positive changes color (default=Green)
colorn	Negative changes color (default=Red)
lag	Lag to comparison (default=1)

Value

Return a vector with colors.

Examples

```
v=c(3,2,5,6,5,4)
p.colorbyvar(x=v,colorp="blue",colorn="grey")

barplot(v,col=p.colorbyvar(v))
```

---

p.col_ord *Ordered bar plot*

Description

p.col_ord make a ordered bar plot.

Usage

```
p.col_ord(
  data,
  xaxis,
  yaxis,
  ybreaks = 10,
  dec = FALSE,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
```

```
  cserie = "black",
  cbserie = cserie,
  cticks = "black",
  lwdserie = 1,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
ybreaks	number of y axis breaks (default=10)
dec	If TRUE serie come be decrescent,if FALSE crescent(default=F)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cserie	color of serie
cbserie	color of serie border (default= same cserie)
cticks	color of axis ticks
lwdserie	size of serie
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a graphic.

Examples

```
v=data.frame("x"=1:5,"y"=c(10,4,8,5,2))
p.col_ord(v,xaxis= v$x,yaxis=v$y)
#or
p.col_ord(v,xaxis= v[[1]],yaxis=v[[2]])

p.col_ord(v,xaxis= v$x,yaxis=v$y,dec=TRUE,percent=FALSE)
p.col_ord(v,xaxis= v$x,yaxis=v$y,dec=TRUE,percent=TRUE)
p.col_ord(v,xaxis= v$x,yaxis=v$y,dec=FALSE,percent=FALSE)
p.col_ord(v,xaxis= v$x,yaxis=v$y,dec=FALSE,percent=TRUE)
```

---

p.col_ord_wl *Ordered bar plot with legend*

Description

p.col_ord_wl make a ordered bar plot with legend.

Usage

```
p.col_ord_wl(
  data,
  xaxis,
  yaxis,
  ybreaks = 10,
  percent = FALSE,
  dec = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cbserie = "black",
  cticks = "black",
  lwdserie = 1,
```

```
  legtitle = "Legend",
  legsize = 8,
  cleg = ctitles,
  legheight = 0.5,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  legpos = "right",
  legdir = "horizontal",
  legcol = "white",
  legspa = 1,
  legvjust = 0.5,
  colors = grDevices::rainbow(length(xaxis), v = 0.7)
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
dec	If TRUE serie come be decrescent,if FALSE crescent(default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cbserie	color of serie border (default= same cserie)
cticks	color of axis ticks
lwdserie	size of serie
legtitle	title of legend box
legsize	size of legend

cleg	color of legend words
legheight	height of legend box
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
legpos	legend position
legdir	legend direction
legcol	color of legend box
legspa	spacement in legend box
legvjust	vertical adjust in legend box
colors	colors of bars, need same number of correspondencies.

Value

Return ordered bar plot with legend.

Examples

```
v=data.frame("x"=1:5,"y"=c(10,4,8,5,2))
p.col_ord_wl(v,xaxis= v$x,yaxis=v$y)
#or
p.col_ord_wl(v,xaxis= v[[1]],yaxis=v[[2]])

p.col_ord_wl(v,xaxis= v$x,yaxis=v$y,dec=TRUE,percent=FALSE)
p.col_ord_wl(v,xaxis= v$x,yaxis=v$y,dec=TRUE,percent=TRUE)
p.col_ord_wl(v,xaxis= v$x,yaxis=v$y,dec=FALSE,percent=FALSE)
p.col_ord_wl(v,xaxis= v$x,yaxis=v$y,dec=FALSE,percent=TRUE)

#Layout example
p.col_ord_wl(v,v$x,v$y,note = "metools - 2020",title = "Layout example",
stitle = "Ordered bar plot",ylab=NULL,wordssize = 10,titlesize = 32,
legspa = 0.5,legvjust = -2.5,legsize = 10,cplot='grey',
cbgrid="black",clgrid= "grey",ctitles = 'white',cleg = 'white',
legcol='black',colors=topo.colors(length(v$x),alpha=0.8))
```

---

p.col_wl *Bar plot with legend*

Description

p.col_wl make a bar plot with legend.

Usage

```
p.col_wl(
  data,
  xaxis,
  yaxis,
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cbserie = "black",
  cticks = "black",
  lwdserie = 1,
  legtitle = "Legend",
  legsize = 8,
  cleg = ctitles,
  legheight = 0.5,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  legpos = "right",
  legdir = "horizontal",
  legcol = "white",
  legspa = 1,
  legvjust = 0.5,
  colors = grDevices::rainbow(length(xaxis), v = 0.7)
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)

ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cbserie	color of serie border (default= same cserie)
cticks	color of axis ticks
lwdserie	size of serie
legtitle	title of legend box
legsize	size of legend
cleg	color of legend words
legheight	height of legend box
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
legpos	legend position
legdir	legend direction
legcol	color of legend box
legspa	spacement in legend box
legvjust	vertical adjust in legend box
colors	colors of bars, need same number of correspondencies.

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame("x"=1:5,"y"=c(10,4,8,5,2))
p.col_wl(v,xaxis= v$x,yaxis=v$y)

p.col_wl(v,xaxis= v$x,yaxis=v$y,colors=c('red','blue','green','grey','yellow'))
```

p.gradientcolor *Create Gradient*

Description

p.gradientcolor is a function to make easy create gradient pallet. Recommended to color graphics created with metools p.functions.

Usage

```
p.gradientcolor(color1, color2, n)
```

Arguments

color1	First gradient color
color2	Last gradient color
n	Number of colors

Value

Return a vector with colors.

Examples

```
p.gradientcolor(color1="white",color2="blue",n=10)

v = p.gradientcolor("white","blue",n=20)
barplot(seq.int(from=1,to=20,by=1),col=v)
```

p.line *Line plot*

Description

p.line make a line plot.

Usage

```
p.line(
  data,
  xaxis,
  yaxis,
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
```

```
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cserie = "black",
  cticks = "black",
  lwdserie = 1,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cserie	color of serie
cticks	color of axis ticks

lwdserie	size of serie
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a line graphic.

Examples

```
v=data.frame("x"=1:5,"y"=c(10,4,8,5,2))
p.line(v,xaxis= v$x,yaxis=v$y)
#or
p.line(v,xaxis= v[[1]],yaxis=v[[2]])
```

---

p.seqdatebreaks *Create Date Interval*

Description

p.seqdatebreaks is a function to break a time axis from graphic in specific interval. This function are recommended to select timeinterval of graphics created with metools p.functions.

Usage

```
p.seqdatebreaks(x, periodicity)
```

Arguments

x	Time data from a Timeserie
periodicity	Time interval (string)

Value

Return a vector with timeinterval.

Examples

```
x <- seq.Date(from=as.Date("2019-01-01"),to=as.Date("2020-01-01"),by=1)
 p.seqdatebreaks(x,periodicity= "2 month")
```

p.tscol *Bar plot in time serie format*

Description

p.tscol make a bar plot in time serie format. The data don't need be a ts object.

Usage

```
p.tscol(
  data,
  xaxis,
  yaxis,
  dateformat = "%Y-%m",
  datebreaks = "1 month",
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cserie = "black",
  cbserie = cserie,
  cticks = "black",
  lwdserie = 1,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data

dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
datebreaks	datebreaks in x axis (default="1 month")
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cserie	color of serie
cbserie	color of serie border (default= same cserie)
cticks	color of axis ticks
lwdserie	size of serie
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a graphic.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(5,3,7,2))

p.tscol(v,v$x,v$y,title="Simple example")

p.tscol(v,v$x,v$y,dateformat="%B",title="Example with colorbyvar",
ylab="Values",xlab=NULL,cserie=p.colorbyvar(v$y))

v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
```

```
to = as.Date('2020-04-01'),by='month'),"y"=c(0.03,-0.05,0.08,-0.02))

p.tscol(v,v$x,v$y,percent=TRUE,title="Example with percent data",xlab=NULL,ylab=NULL)

p.tscol(v,v$x,v$y,percent=TRUE,yaccuracy=1,title="y accuracy set",xlab=NULL,ylab=NULL)

p.tscol(v,v$x,v$y,percent=TRUE,yaccuracy=1,title="Example with colorbypositive",xlab=NULL,ylab=NULL,
cserie=p.colorbypositive(v$y),cbserie="black",lwdserie=1) #lwdserie change the board in this case
```

`p.tsl` *Line plot in time serie format*

Description

p.tsl make a line plot in time serie format. The data don't need be a ts object.

Usage

```
p.tsl(
  data,
  xaxis,
  yaxis,
  dateformat = "%Y-%m",
  datebreaks = "1 month",
  ybreaks = 10,
  percent = FALSE,
  yaccuracy = 0.01,
  ydecimalmark = ".",
  title = "Title",
  xlab = "X axis",
  ylab = "Y axis",
  stitle = NULL,
  note = NULL,
  ctitles = "black",
  cscales = ctitles,
  cbgrid = "white",
  clgrid = cbgrid,
  cplot = "white",
  cserie = "black",
  cticks = "black",
  lwdserie = 1,
  pnote = 1,
  cbord = cplot,
  titlesize = 20,
  wordssize = 12,
  snote = 11,
  xlim = NULL
)
```

Arguments

data	a dataframe
xaxis	x axis data
yaxis	y axis data
dateformat	format of date in x axis (need a dataformat string) (default ="%Y-%m")
datebreaks	datebreaks in x axis (default="1 month")
ybreaks	number of y axis breaks (default=10)
percent	If TRUE y axis in percent (default=F)
yaccuracy	a round for y axis (default=0.01)
ydecimalmark	y decimal mark (default=".")
title	title of plot
xlab	x axis label
ylab	y axis label
stitle	subtitle
note	note
ctitles	color of titles (title,xlab,ylab)
cscales	color of the scales (default= same ctitles)
cbgrid	color of grid background
clgrid	color of grid lines
cplot	color of plot background
cserie	color of serie
cticks	color of axis ticks
lwdserie	size of serie
pnote	position of note (default=1) (only numbers)
cbord	color of plot border (default= same cplot)
titlesize	size of title (default=20) (only numbers)
wordssize	size of words (default=12) (only numbers)
snote	size of note (default=11) (only numbers)
xlim	limit of x axis (default=NULL)

Value

Return a dataframe with transformed columns.

Examples

```
v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(5,3,7,2))

p.tsl(v,v$x,v$y,title="Simple example")

v=data.frame("x"=seq.Date(as.Date('2020-01-01'),
to = as.Date('2020-04-01'),by='month'),"y"=c(0.03,-0.05,0.08,-0.02))

p.tsl(v,v$x,v$y,percent=TRUE,title="Example with percent data",xlab=NULL,ylab=NULL)

p.tsl(v,v$x,v$y,percent=TRUE,yaccuracy=1,title="y accuracy set",xlab=NULL,ylab=NULL)
```

---

pct_change *Percentual change*

Description

pct_change calculate the percentual change in t periods of a serie. We can use this function to calculate the acumulated variation of an index, for example to calculate the accumulated variation in 12 months just set t parameter to 12

Usage

```
pct_change(data, colnum, t = nrow(data[colnum]) - 1, nafill = NA)
```

Arguments

data	a dataframe
colnum	number of column
t	number of periods to accumulate (default= number of rows)
nafill	set value to fill NA's before first t value

Value

Return a dataframe.

Examples

```
v=data.frame(test=c(1,2,3,4,5,6,7,8,9,10,11,12,13))
pct_change(v)
```

stattable *Descritive statistic table*

Description

stattable make a descritive statistic table.

Usage

```
stattable(data, horiz = FALSE, translate = FALSE)
```

Arguments

data	a dataframe
horiz	defines table be a horizontal table (default=FALSE)
translate	if TRUE translate table to PT-BR (default=FALSE)

Value

Return a dataframe with descritive statistics.

Examples

```
v=data.frame(dataone=c(3,2,5,6,5,4),datatwo=c(33,22,55,66,55,44)
,datathree=c(133,122,155,166,155,144))
stattable(v) #vertical table
stattable(v,translate=TRUE) #vertical table translated
stattable(v,horiz=TRUE) #horizontal table
stattable(v,horiz=TRUE,translate=TRUE) #horizontal table translated
```

Index