

# Package ‘FinAna’

January 20, 2025

**Type** Package

**Title** Financial Analysis and Regression Diagnostic Analysis

**Version** 0.1.2

**Author** Xuanhua(Peter) Yin <peteryin.sju@hotmail.com>

**Maintainer** Xuanhua(Peter) Yin <peteryin.sju@hotmail.com>

**Description** Functions for financial analysis and financial modeling, including batch graphs generation, beta calculation, descriptive statistics, annuity calculation, bond pricing and financial data download.

**Note** Few parts are still preliminary and might be changed in the near future. And more functions will be add as easier tools to higher efficiency in analyzing.

**License** GPL (>= 2)

**LazyData** TRUE

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2017-10-26 03:08:49 UTC

## Contents

annu.fv . . . . .	2
annu.pv . . . . .	2
annu.pv.df . . . . .	3
betaf . . . . .	4
bond.price . . . . .	4
corm . . . . .	5
desc . . . . .	5
get.mode . . . . .	6
get.price.google . . . . .	6
get.price.yahoo . . . . .	7
kur . . . . .	7

ploth . . . . .	8
ploths . . . . .	8
plotsm . . . . .	9
plotts . . . . .	9
rr . . . . .	10
sk . . . . .	10

<b>Index</b>	<b>11</b>
--------------	-----------

---

annu.fv	<i>Calculate future value of annuity</i>
---------	--

---

### Description

Calculate future value of an ordinary annuity or an annuity due.

### Usage

```
annu.fv(pmt,i,n,type = 0)
```

### Arguments

pmt	:the equal amount of payment of each period
i	:interest rate according to the period
n	:number of periods
type	:type = 0 for ordinary annuity, type = 1 for annuity due

### Examples

```
#annu.fv(100,0.0248,10,0)
```

---

annu.pv	<i>Calculate present value of annuity</i>
---------	---

---

### Description

Calculate present value of an ordinary annuity or an annuity due.

### Usage

```
annu.pv(pmt,i,n,type = 0)
```

**Arguments**

- pmt :the equal amount of payment of each period
- i :interest rate according to the period
- n :number of periods
- type :type = 0 for ordinary annuity, type = 1 for annuity due

**Examples**

```
#annu.pv(100,0.0248,10,0)
```

---

<code>annu.pv.df</code>	<i>Calculate present value of annuity</i>
-------------------------	---

---

**Description**

Calculate present value of an ordinary annuity or an annuity due.

**Usage**

```
annu.pv.df(pmt,i,n,k)
```

**Arguments**

- pmt :the equal amount of payment of each period
- i :interest rate according to the period
- n :number of periods
- k :number of periods deferred

**Examples**

```
#annu.pv(100,0.0248,10,4,0)
```

---

betaf	<i>Calculating beta for a company or a select of companies</i>
-------	--

---

**Description**

Calculating beta using common method or linear regression(OLS)

**Usage**

```
betaf(x,y,method)
```

**Arguments**

x	:a vector or a data.frame of rate of return of companies
y	:name of the independent variable
method	:method of calculation; method = 1 for a common expression of beta(see detail); method = 2 using linear regression to estimate the beta

**Examples**

```
#betaf(appl,sp500)
```

---

bond.price	<i>Calculate the plain vanilla bond price</i>
------------	---

---

**Description**

Calculate the plain vanilla bond price

**Usage**

```
bond.price(par,c,n,yield,m)
```

**Arguments**

par	:the face value of the bond
c	:the annual coupon rate of the bond
n	:number of years
yield	:the annual yield to maturity of a bond
m	:couponding period in a year

**Examples**

```
#bond.price(1000,0.03,10,0.0248,2)
```

---

`corm`*Correlation matrix and correlation ranking of a data.frame*

---

**Description**

Calculating the descriptive statistics of a data.frame and exporting in a data.frame

**Usage**

```
corm(x,n)
```

**Arguments**

x                    :a data.frame  
n                    :number of decimal points

**Examples**

```
#corm(sp1500,3) for correlation matrix of sp1500
```

---

`desc`*Descriptice statistics of a data.frame*

---

**Description**

Calculating the descriptive statistics of a data.frame and exporting in a data.frame

**Usage**

```
desc(x,n)
```

**Arguments**

x                    :a data.frame  
n                    :number of decimal points

**Examples**

```
#desc(sp1500,3) for descriptive statistics of sp1500
```

---

get.mode *Calculating mode for numeric data*

---

### Description

Calculating mode for numeric data

### Usage

```
get.mode(x)
```

### Arguments

x :a numeric variable(vector)

### Examples

```
# get.mode(return)
```

---

get.price.google *Download financial data from google finance*

---

### Description

Download stock prices for one company or a list of companies from google finance. And further application of rate of return function and beta function in the package for more analysis.

### Usage

```
get.price.google(tkr, bg = "2001-01-01",ed = "today")
```

### Arguments

tkr :company ticker, e.g. "BABA","AMZN"  
 bg :beginning date, e.g."2000-02-29"  
 ed :ending date, e.g. "today", "2016-11-10"

### Examples

```
#get.price.google("GOOG")
#get.price.google("GOOG", bg = "2001-01-01",ed = "today")
# the two above are the same
#
# tkr <- c("AAPL", "IBM","YHOO")
# pricelist <- get.price.google(tkr, bg = "2001-01-01",ed = "today")
# aapl <- pricelist[1] # convert to single data.frame
# ibm <- pricelist[2] # convert to single data.frame
# yhoo <- pricelist[3] # convert to single data.frame
```

---

get.price.yahoo      *Download financial data from Yahoo finance*

---

### Description

Download stock prices for one company or a list of companies from Yahoo finance. The function can download daily, weekly and monthly data. And further application of rate of return function and beta function in the package for more analysis.

### Usage

```
get.price.yahoo(tkr, bg = "first", ed = "today", f = "d")
```

### Arguments

tkr                    :company ticker, e.g. "BABA","AMZN"  
bg                    :beginning date, e.g. "first","2000-02-29"  
ed                    :ending date, e.g. "today", "2016-11-10"  
f                     :frequency, e.g. "d" for daily,"w" for weekly,"m" for monthly

### Examples

```
#get.price.yahoo("GOOG")  
#get.price.yahoo("GOOG", bg = "first", ed = "today", f = "d")  
# the two above are the same  
#  
# tkr <- c("AAPL", "IBM", "YHOO")  
# pricelist <- get.price.yahoo(tkr, bg = "first", ed = "today", f = "m")  
# aapl <- pricelist[1] # convert to single data.frame  
# ibm <- pricelist[2] # convert to single data.frame  
# yhoos <- pricelist[3] # convert to single data.frame
```

---

kur                    *Calculating kurtosis for numeric data*

---

### Description

Kurtosis

### Usage

```
kur(x)
```

### Arguments

x                    :a numeric variable

**Examples**

```
#kur(return) for skewness of variable return
```

---

```
ploth
```

*Plot histograms for a data.frame*

---

**Description**

Plotting histograms for a data.frame. Also the function will name the graphs and number the graphs.

**Usage**

```
ploth(x,c,l)
```

**Arguments**

```
x
```

:a dataframe

```
c
```

:is there dummy variable in the data.frame; c = 0 when there is none; c = 1 when there is

```
l
```

: number of labeling starts at (default = 1)

**Examples**

```
#ploth(sp500,0,20) for histograms of sp500 which does not has dummy variables
```

---

```
ploths
```

*Plot histograms and scatter plots for a data.frame*

---

**Description**

Plotting histograms or scatter plots of your choice for a data.frame. Also the function will name the graphs and number them. The purpose of the function is to save time when plotting graphs for a regression analysis or other usage. The function can plot, name and number the graphs at one step.

**Usage**

```
ploths(x,a,dependent,c,l)
```

**Arguments**

```
x
```

:a dataframe

```
a
```

:the type of graph you want; a = 1 for histograms; a = 2 for scatter plots; a = 0 for both

```
dependent
```

:the dependent variable for scatterplots

```
c
```

:is there dummy variable in the dataframe; c = 0 when there is none; c = 1 when there is

```
l
```

: number of labeling starts at (default = 1)



**Examples**

```
#ploths(sp500,0,"price",0,20)
```

---

plotsm	<i>Plot scatter smooth plots for a data.frame</i>
--------	---

---

**Description**

Plotting scatter smooth plots for a data.frame, with name, number and labels.

**Usage**

```
plotsm(x,dependent,c,l)
```

**Arguments**

x	:a dataframe
dependent	:the dependent variable
c	:is there dummy variable in the data.frame; c = 0 when there is none; c = 1 when there is
l	: number of labeling starts at (default = 1)

**Examples**

```
# plotsm(JPM-ratios,"price",0,20)
```

---

plotts	<i>Plot time series plots for a data.frame</i>
--------	--

---

**Description**

Plotting time series plots for a data.frame, with name the graphs and number the graphs.

**Usage**

```
plotts(x,c,l)
```

**Arguments**

x	:a dataframe
c	:is there dummy variable in the data.frame; c = 0 when there is none; c = 1 when there is
l	: number of labeling starts at (default = 1)

**Examples**

```
#plotts(sp500,0,20)
```

---

rr *Calculating rate of return of a vector*

---

**Description**

Calculating the rate of return of a vector for further analysis, including calculating beta of companies, plotting to see the trend of the stock for technical analysis

**Usage**

```
rr(x, n)
```

**Arguments**

x :a vector of company prices  
n : number of lags

**Examples**

```
#rr(aapl, 1)
```

---

sk *Calculating skewness for numeric data*

---

**Description**

Calculating Pearson's skewness in three types: mode, median, and mean

**Usage**

```
sk(x, type = 3)
```

**Arguments**

x :a numeric variable  
type :type = 1 for mode skewness; type = 2 for median skewness; type = 3 for mean skewness

**Examples**

```
#sk(return) for skewness of variable return
```

# Index

`annu.fv`, 2  
`annu.pv`, 2  
`annu.pv.df`, 3

`betaf`, 4  
`bond.price`, 4

`corm`, 5

`desc`, 5

`get.mode`, 6  
`get.price.google`, 6  
`get.price.yahoo`, 7

`kur`, 7

`ploth`, 8  
`ploths`, 8  
`plotsm`, 9  
`plotts`, 9

`rr`, 10

`sk`, 10