COMPUTATIONAL FINANCE & RISK MANAGEMENT

UNIVERSITY *of* WASHINGTON

Department of Applied Mathematics

Financial Data Modeling and Analysis in R

Guy Yollin

Applied Mathematics
University of Washington

Outline

1. Asset return calculations
2. Computing asset returns in R

Lecture references

D. Ruppert.
Statistics and Data Analysis for Financial Engineering.
Springer, 2010.

- Chapter 2 - Returns

Outline

1. Asset return calculations
2. Computing asset returns in R

Net Returns

Analysis of financial data widely employs the use of asset returns

Let P_t be the price of a non-dividend paying asset at time t

- Net Return

$$R_t = \frac{P_t}{P_{t-1}} - 1 = \frac{P_t - P_{t-1}}{P_{t-1}} = \%\Delta P_t$$

- Gross Return

$$1 + R_t = \frac{P_t}{P_{t-1}}$$

Log Returns

- Log Return (*aka* continuously compounded return)

$$r_t = \ln(1 + R_t) = \ln\left(\frac{P_t}{P_{t-1}}\right) = \ln(P_t) - \ln(P_{t-1})$$

$$r_t = p_t - p_{t-1} \qquad \text{where } p_t = \ln(P_t)$$

- r_t is the continuously compounded growth rate in prices

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

$$e^{r_t} = \frac{P_t}{P_{t-1}}$$

$$P_t = P_{t-1}e^{r_t}$$

Comparison of net returns and log returns

Net returns and log returns should be very similar for small returns (less than about 10%):

$$R_t \approx r_t \text{ for small } R_t$$

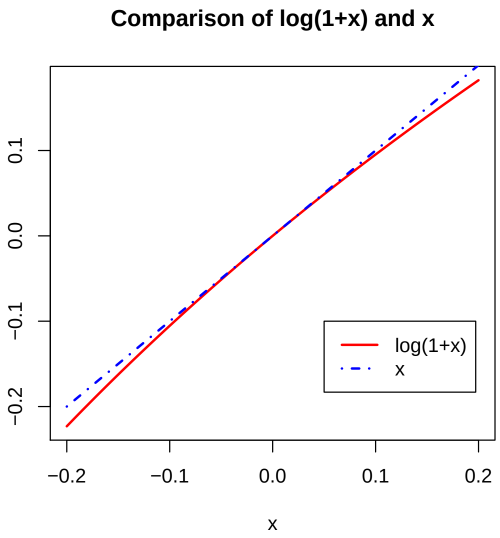
```
log( 1 + c(0.1, 0.05, 0.01, 0, -0.01, -0.05, -0.1) )

## [1]  0.09531  0.04879  0.00995  0.00000 -0.01005 -0.05129 -0.10536

x <- seq(-0.2,0.2,len=100)
x1 <- log(1+x)
plot(x=x, y=x1, xlab="x", ylab="", lwd=2, type="l", col=2)
lines(x=x, y=x, col=4, lty=4, lwd=2)
title(main="Comparison of log(1+x) and x")
legend(x=0.05, y=-0.1, c("log(1+x)","x"), col=c(2, 4), lty = c(1, 4), lwd=2)
```

`type="l"`	plot lines
`lwd`	line width
`lty`	line type

Compare net returns and log returns



SDAFE Fig 2.1

Multi-Period Returns

- k-period Net Return

$$R_t(k) = \prod_{j=0}^{k-1}(1 + R_{t-j}) - 1$$

- k-period Log Return

$$r_t(k) = \sum_{j=0}^{k-1} r_{t-j}$$

The additivity of continuously compounded returns to form multiperiod returns is an important property for statistical modeling

Lognormal price process

- Single period log return

$$r_t \sim \text{i.i.d. } N(\mu, \sigma^2)$$

- Multiperiod log return

$$r_t(k) \sim N(\mu k, \sigma^2 k)$$

- Process for the log-price (normal random walk with drift)

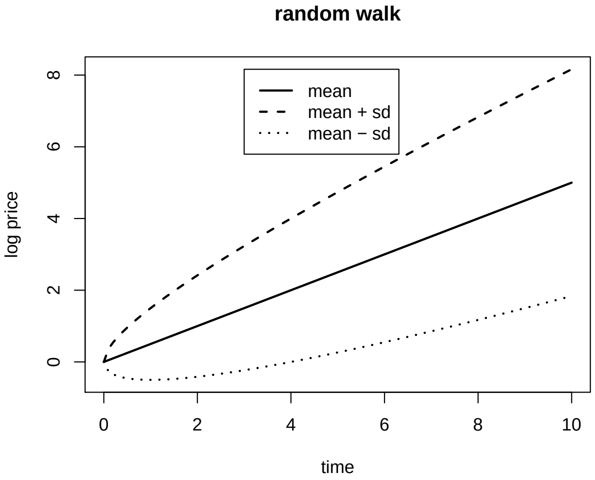
$$p_T = p_0 + \mu T + \sum_{t=1}^{T} \varepsilon_t, \quad \varepsilon_t \sim \text{i.i.d. } N(0, \sigma^2)$$

$$p_T \sim N(p_0 + \mu T, \sigma^2 T), \quad \mathrm{SD}(p_T) = \sqrt{T}\sigma$$

- Process for the asset price (lognormal geometric random walk)

$$P_T = e^{p_T} = P_0 e^{\mu T + \sum_{t=1}^{T} \varepsilon_t} = P_0 e^{\mu T} e^{\sum_{t=1}^{T} \varepsilon_t}$$

Random walk probability cone



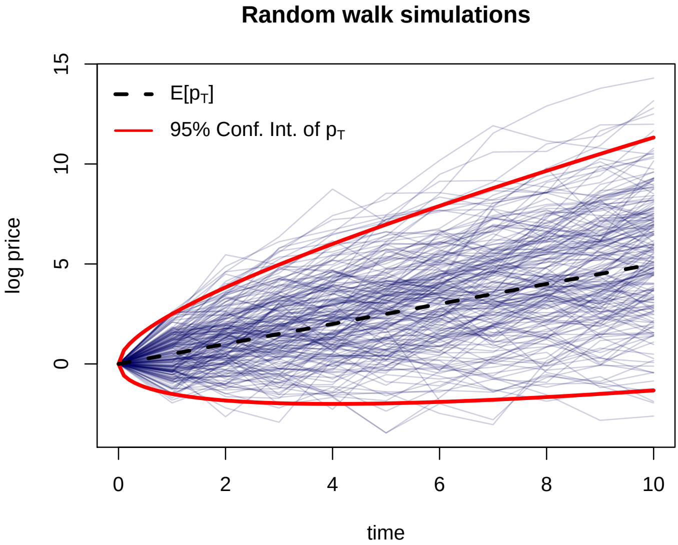
SDAFE Fig 2.2

Random walk probability cone

```
S0 <- 0
mu <- 0.5
sigma <- 1
tm <- seq(0, 10, len=100)
S <- S0 + mu*tm
ubound <- S0 + mu*tm + sigma*sqrt(tm)
lbound <- S0 + mu*tm - sigma*sqrt(tm)
ylim = range(c(ubound, lbound))
plot(x=tm, y=S, ylim=ylim, xlab="time", ylab="log price", main="random walk",
  type="l", lwd=2)
lines(tm, ubound, lty=2, lwd=2)
lines(tm, lbound, lty=3, lwd=2)
legend(3, max(ylim), c("mean","mean + sd","mean - sd"), lty = c(1,2,3), lwd=2)
```

- note application of the *square-root of time rule*

Random walk probability cone



Random walk probability cone

```
set.seed(2)
nsim <- 200

mat <- matrix( rnorm(n=10*nsim, mean=mu, sd=sigma), nrow=nsim, ncol=10 )
mat <- cbind(0, mat)
mat1 <- apply(mat, 1, cumsum)

matplot(x=0:10, y=mat1, type="l",
  col=rgb(0,0,100,50,maxColorValue=255),
  lty=1, xlab="time", ylab="log price")

ub <- S0 + mu*tm + 2*sigma*sqrt(tm)
lb <- S0 + mu*tm - 2*sigma*sqrt(tm)
lines(x=tm, y=ub, col=2, lwd=3)
lines(x=tm, y=lb, col=2, lwd=3)
lines(x=tm, y=S, type="l", col=1, lty=2, lwd=3)

l.str1 <- expression("E["*p[T]*"]")
l.str2 <- expression("95% Conf. Int. of "*p[T])
legend(x="topleft", legend=c(l.str1,l.str2), col=c(1,2), lty=c(2,1),
  bty="n", lwd=c(3,2), y.intersp=1.5, cex=1.0)

title("Random walk simulations")
```

Outline

1. Asset return calculations
2. Computing asset returns in R

Simple returns from a vector of prices

$$R_t = \frac{P_t}{P_{t-1}} - 1 = \frac{P_t - P_{t-1}}{P_{t-1}} = \%\Delta P_t$$

```
P <- c(265.50, 264.27, 266.49, 253.81, 269.20, 277.69, 301.22, 280.98, 312.64,
  364.03, 393.62, 398.79)

P[-length(P)]

##  [1] 265.5 264.3 266.5 253.8 269.2 277.7 301.2 281.0 312.6 364.0 393.6

P[-1]

##  [1] 264.3 266.5 253.8 269.2 277.7 301.2 281.0 312.6 364.0 393.6 398.8

(R <- P[-1] / P[-length(P)] - 1)

##  [1] -0.004633  0.008400 -0.047582  0.060636  0.031538  0.084735 -0.067193
##  [8]  0.112677  0.164374  0.081285  0.013134
```

Simple returns from a vector of prices

```
args(diff.default)

## function (x, lag = 1L, differences = 1L, ...)
## NULL

diff(P)

##  [1]  -1.23   2.22 -12.68  15.39   8.49  23.53 -20.24  31.66  51.39  29.59
## [11]   5.17

(R <- diff(P) / P[-length(P)])

##  [1] -0.004633  0.008400 -0.047582  0.060636  0.031538  0.084735 -0.067193
##  [8]  0.112677  0.164374  0.081285  0.013134
```

- diff is a generic function for lagged differences

Log returns from a vector of prices

$$r_t = \ln(1 + R_t) = \ln\left(\frac{P_t}{P_{t-1}}\right) = \ln(P_t) - \ln(P_{t-1}) = p_t - p_{t-1}$$

```
log(1+R)

##  [1] -0.004644  0.008365 -0.048751  0.058869  0.031051  0.081336 -0.069557
##  [8]  0.106769  0.152184  0.078150  0.013049

(r <- diff(log(P)))

##  [1] -0.004644  0.008365 -0.048751  0.058869  0.031051  0.081336 -0.069557
##  [8]  0.106769  0.152184  0.078150  0.013049

exp(r) - 1

##  [1] -0.004633  0.008400 -0.047582  0.060636  0.031538  0.084735 -0.067193
##  [8]  0.112677  0.164374  0.081285  0.013134
```

Simple returns from a zoo object

```
library(zoo)
(z <- zooreg(P, as.yearmon("2013-01"), freq = 12))

## Jan 2013 Feb 2013 Mar 2013 Apr 2013 May 2013 Jun 2013 Jul 2013 Aug 2013
##    265.5    264.3    266.5    253.8    269.2    277.7    301.2    281.0
## Sep 2013 Oct 2013 Nov 2013 Dec 2013
##    312.6    364.0    393.6    398.8

class(z)

## [1] "zooreg" "zoo"

(R.z <- z[-1] / z[-length(z)] - 1)

## Feb 2013 Mar 2013 Apr 2013 May 2013 Jun 2013 Jul 2013 Aug 2013 Sep 2013
##        0        0        0        0        0        0        0        0
## Oct 2013 Nov 2013
##        0        0
```

- Arithmetic between vectors is element-by-element
- Arithmetic between time series objects is timestamp-by-timestamp

Simple returns from a zoo object

```
args(getS3method("lag","zoo"))

## function (x, k = 1, na.pad = FALSE, ...)
## NULL

R

##  [1] -0.004633  0.008400 -0.047582  0.060636  0.031538  0.084735 -0.067193
##  [8]  0.112677  0.164374  0.081285  0.013134

(R.z <- diff(z) / lag(z,-1))

##  Feb 2013  Mar 2013  Apr 2013  May 2013  Jun 2013  Jul 2013  Aug 2013
## -0.004633  0.008400 -0.047582  0.060636  0.031538  0.084735 -0.067193
##  Sep 2013  Oct 2013  Nov 2013  Dec 2013
##  0.112677  0.164374  0.081285  0.013134
```

- To shift a zoo object back in time, `lag=-1` (not the default)

Log returns from a zoo object

```
r
```

```
##  [1] -0.004644  0.008365 -0.048751  0.058869  0.031051  0.081336 -0.069557
##  [8]  0.106769  0.152184  0.078150  0.013049
```

```
(r.z <- diff(log(z)))
```

```
##  Feb 2013  Mar 2013  Apr 2013  May 2013  Jun 2013  Jul 2013  Aug 2013
## -0.004644  0.008365 -0.048751  0.058869  0.031051  0.081336 -0.069557
##  Sep 2013  Oct 2013  Nov 2013  Dec 2013
##  0.106769  0.152184  0.078150  0.013049
```

- `diff(log(price))` works for vectors and time series

Simple returns from an xts object

```
library(xts)
(x <- as.xts(z))

##          [,1]
## Jan 2013 265.5
## Feb 2013 264.3
## Mar 2013 266.5
## Apr 2013 253.8
## May 2013 269.2
## Jun 2013 277.7
## Jul 2013 301.2
## Aug 2013 281.0
## Sep 2013 312.6
## Oct 2013 364.0
## Nov 2013 393.6
## Dec 2013 398.8

class(x)

## [1] "xts" "zoo"
```

Simple returns from an xts object

```
args(lag.xts)

## function (x, k = 1, na.pad = TRUE, ...)
## NULL

(R.x <- diff(x) / lag(x))

##                [,1]
## Jan 2013         NA
## Feb 2013 -0.004633
## Mar 2013  0.008400
## Apr 2013 -0.047582
## May 2013  0.060636
## Jun 2013  0.031538
## Jul 2013  0.084735
## Aug 2013 -0.067193
## Sep 2013  0.112677
## Oct 2013  0.164374
## Nov 2013  0.081285
## Dec 2013  0.013134
```

- To shift an xts object back in time, `lag=1` (the default)

Log returns from an xts object

```
r

## [1] -0.004644  0.008365 -0.048751  0.058869  0.031051  0.081336 -0.069557
## [8]  0.106769  0.152184  0.078150  0.013049

(r.x <- diff(log(x)))

##               [,1]
## Jan 2013        NA
## Feb 2013 -0.004644
## Mar 2013  0.008365
## Apr 2013 -0.048751
## May 2013  0.058869
## Jun 2013  0.031051
## Jul 2013  0.081336
## Aug 2013 -0.069557
## Sep 2013  0.106769
## Oct 2013  0.152184
## Nov 2013  0.078150
## Dec 2013  0.013049
```

- `diff(log(price))` works for vectors and time series

COMPUTATIONAL FINANCE & RISK MANAGEMENT

UNIVERSITY *of* WASHINGTON

Department of Applied Mathematics

http://depts.washington.edu/compfin