STAT 7260 Project Code

Ke Wang

2023-11-15

```
# set start and end date of data to download
dateStart <- "2013-01-01"
dateEnd <- "2023-7-31"

# Bank of Montreal
bmo <- get.hist.quote(instrument = "BMO", start = dateStart, end = dateEnd, quote = c("AdjClose"),
                      retclass = "zoo")
```

Historical Finance Data Download from Yahoo!Finance

```
## time series starts 2013-01-02
## time series ends   2023-07-28
```

```
# Royal Bank of Canada
rbc <- get.hist.quote(instrument = "RBC", start = dateStart, end = dateEnd, quote = c("AdjClose"),
                      retclass = "zoo")
```

```
## time series starts 2013-01-02
## time series ends   2023-07-28
```

```
# Toronto-Dominion Bank
td <- get.hist.quote(instrument = "TD", start = dateStart, end = dateEnd, quote = c("AdjClose"),
                     retclass = "zoo")
```

```
## time series starts 2013-01-02
## time series ends   2023-07-28
```

```
# Bank of Nova Scotia
bns <- get.hist.quote(instrument = "BNS", start = dateStart, end = dateEnd, quote = c("AdjClose"),
                      retclass = "zoo")
```

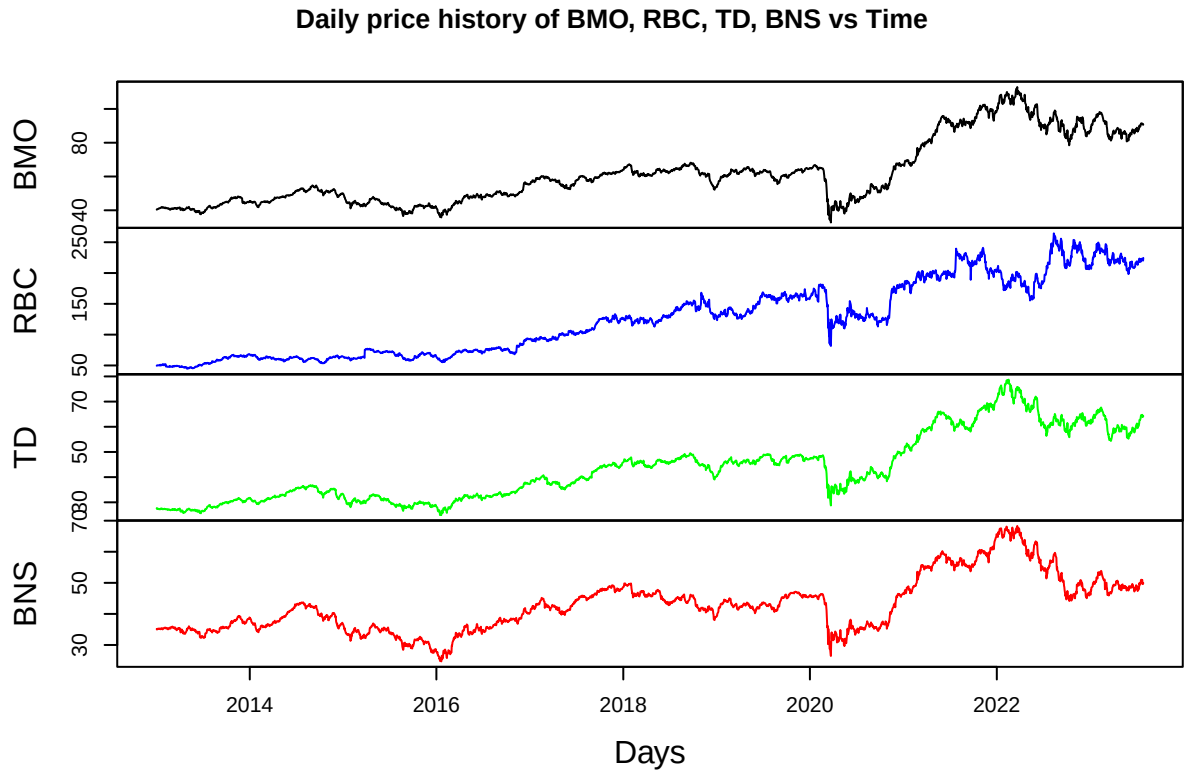
```
## time series starts 2013-01-02
## time series ends   2023-07-28
```

```
# preview of datasets
data <- merge(bmo, rbc, td, bns) # price data
head(data)
```

```
##            Adjusted.bmo Adjusted.rbc Adjusted.td Adjusted.bns
## 2013-01-02     40.39813     49.50009    27.66907     35.16365
## 2013-01-03     40.39163     50.05095    27.36091     34.96489
## 2013-01-04     40.62583     50.12827    27.50843     35.16365
## 2013-01-07     40.74943     50.25390    27.45270     35.22387
## 2013-01-08     40.78847     50.17658    27.45926     35.26003
## 2013-01-09     40.95759     50.24424    27.33140     35.13956
```

```
save(data, file="stock.Rdata")
```

```
# plot of daily price
names(data) <- c("BMO", "RBC", "TD", "BNS")
plot(data, xlab = "Days", main = "Daily price history of BMO, RBC, TD, BNS vs Time", col = c("black","B
```



EDA

```
# calculation of continuously compounded returns as difference in log prices
return.cc <- diff(log(data))

# remove missing values
sum(is.na(return.cc))
```

```
## [1] 0
```

```
return.cc <- na.omit(return.cc)
sum(is.na(return.cc))
```

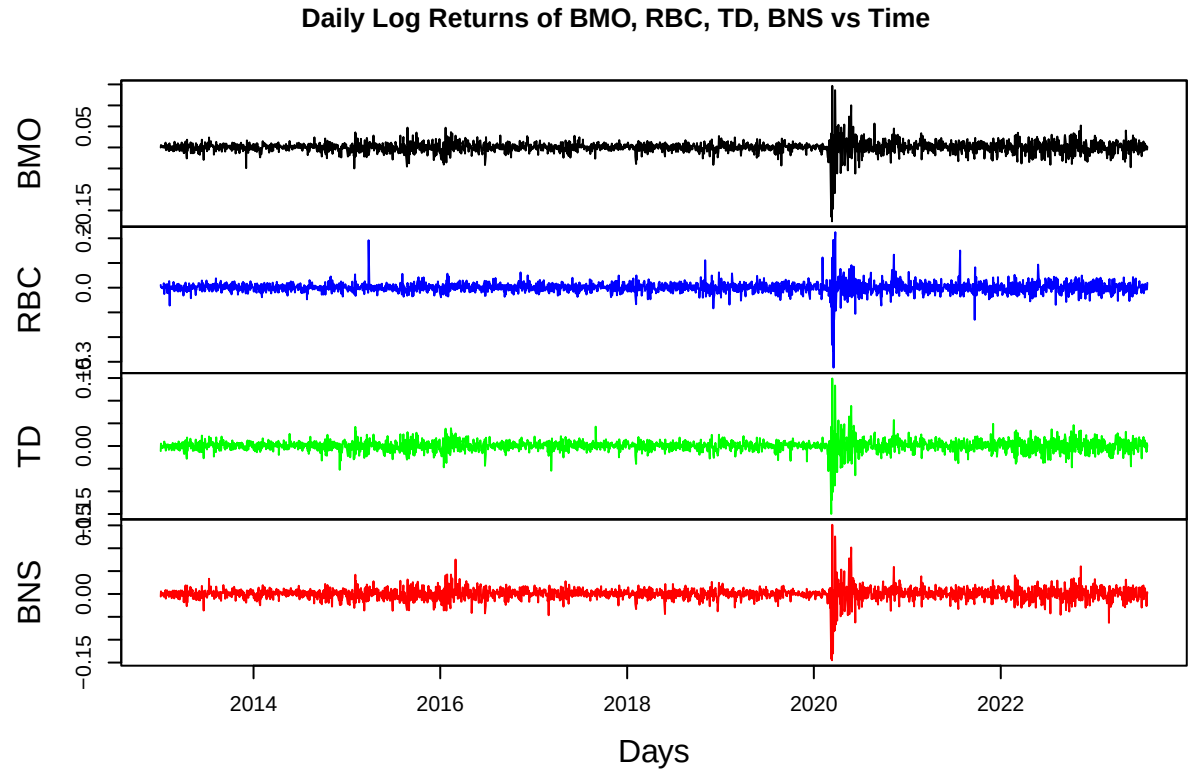
```
## [1] 0
```

```
head(return.cc)
```

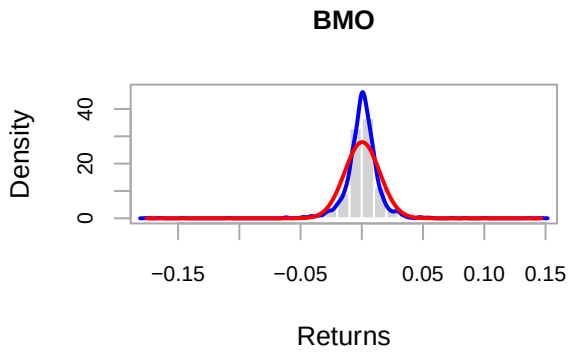
```
##                         BMO           RBC            TD           BNS
## 2013-01-03 -0.0001610120  0.0110671027 -0.0111995123 -0.005668709
## 2013-01-04  0.0057814745  0.0015434835  0.0053768749  0.005668709
## 2013-01-07  0.0030377810  0.0025031002 -0.0020277316  0.001711065
## 2013-01-08  0.0009575821 -0.0015396979  0.0002386966  0.001025928
## 2013-01-09  0.0041378103  0.0013474032 -0.0046670559 -0.003422305
```

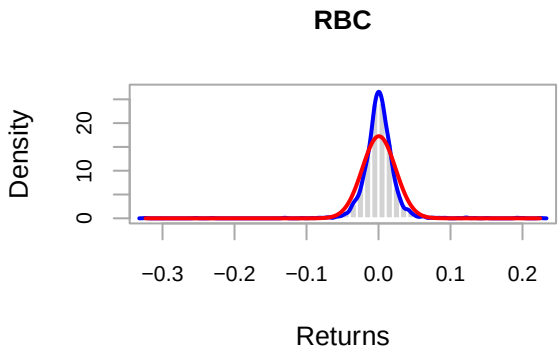
```
## 2013-01-10  0.0034884361  0.0001922947 -0.0003600907  0.001541210
```

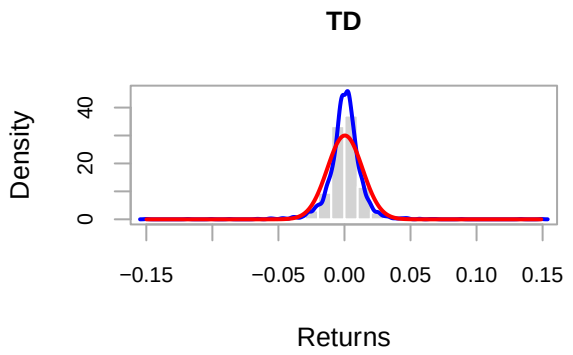
```r
# plot of log returns
plot(return.cc, xlab = "Days", main = "Daily Log Returns of BMO, RBC, TD, BNS vs Time", col = c("black"
```

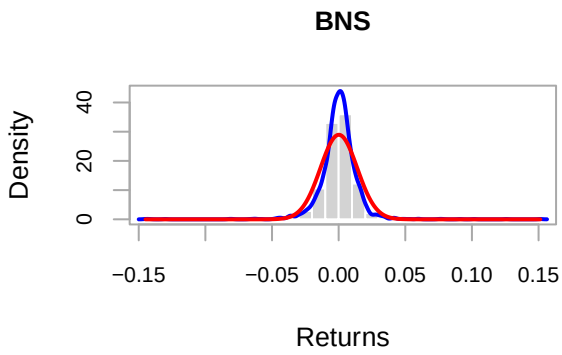



```r
# density and histograms of log returns
names(return.cc) <- c("BMO", "RBC", "TD", "BNS")
par(mfrow=c(2,2))
for (i in 1:ncol(return.cc)) {
  chart.Histogram(return.cc[,i], main = paste(names(return.cc)[i]), breaks = 40,
  colorset= c("lightgray","blue","red"), methods = c("add.density", "add.normal"))
}
```










```
return.cc <- data.frame(return.cc)

mean.cc <- apply(return.cc, 2, mean)
sd.cc <- apply(return.cc, 2, sd)
min.cc <- apply(return.cc, 2, min)
max.cc <- apply(return.cc, 2, max)
kurt.cc <- kurtosis(return.cc)
skew.cc <- skewness(return.cc)

ss1 <- data.frame(mean.cc, sd.cc, min.cc, max.cc, t(kurt.cc), t(skew.cc))
```

```
# correlation of return series
acf1 <- apply(return.cc, 2, acf, lag=1, plot=F)
corr1 <- c(acf1$BMO$acf[2], acf1$RBC$acf[2], acf1$TD$acf[2], acf1$BNS$acf[2]) # extracting acf values

# correlation of squared return series
acf2 <- apply(return.cc^2, 2, acf, lag=1, plot=F)
corr2 <- c(acf2$BMO$acf[2], acf2$RBC$acf[2], acf2$TD$acf[2], acf2$BNS$acf[2])

# correlation of absolute return series
acf3 <- apply(abs(return.cc), 2, acf, lag=1, plot=F)
corr3 <- c(acf3$BMO$acf[2], acf3$RBC$acf[2], acf3$TD$acf[2], acf3$BNS$acf[2])
```

Basic Statistics

```
# function to get optimal alpha that gives minimum SSE
alpha <- seq(.01,.3,.01)
alpha.opt.e <- c()
SSE <- matrix(0, length(alpha), ncol(return.cc))

for (i in 1:length(alpha)) {
  for (j in 1:ncol(return.cc)) {
    s <- mean(return.cc[,j]^2)
    for (k in 1:nrow(return.cc)) {
      error <- return.cc[k,j]^2 - s
      SSE[i,j] <- error^2 + SSE[i,j]
      s.k <- alpha[i]*(return.cc[k,j]^2) + (1 - alpha[i])*s
      s <- s.k
    }
     alpha.opt.e[j] <- alpha[which.min(SSE[,j])]
  }
}
alpha.opt.e
```

Method 1: EWMA

```
## [1] 0.25 0.30 0.21 0.25
```

```
# find one-step ahead forward volatility forecast using optimal alpha
v.e <- c()
for (i in 1:ncol(return.cc)) {
  s <- mean(return.cc[,i]^2)
  for (j in 1:nrow(return.cc)) {
    s.j <- alpha.opt.e[i]*(return.cc[j,i]^2) + (1 - alpha.opt.e[i])*s
    s <- s.j
  }
  v.e[i] <- sqrt(s)
}
v.e
```

```
## [1] 0.004441767 0.012196756 0.007406026 0.010039571
```

```
# find VaR forecast
p <- 0.01
VaR.e <- (-1)*qnorm(p)*v.e
VaR.e
```

```
## [1] 0.01033310 0.02837390 0.01722899 0.02335553
```

```
# function to compute sign correlation
rho.cal <- function(X){
  rho.hat <- cor(sign(X-mean(X)), X-mean(X))
  return(rho.hat)
}

# find sign correlation
rho.sign <- NULL
for (i in 1:ncol(return.cc)) {
  rho.sign[i] <- apply(return.cc[i], 2, rho.cal)
```

```
}
rho.sign
```

DDEWMA

```
## [1] 0.6298253 0.6281500 0.6480272 0.6442742
```

```
# find degrees of freedom
df <- NULL
for (i in 1:ncol(return.cc)) {
  fun <- function (x) rho.sign[i]*(x-1)*beta(x/2,1/2)-2*sqrt(x-2)
  df[i] <- uniroot(fun, c(2, 8), extendInt = "yes")$root
}
df
```

```
## [1] 2.946908 2.934449 3.099599 3.065250
```

```
# function to get optimal alpha that gives minimum SSE
alpha.opt.d <- c()
SSE <- matrix(0, length(alpha), ncol(return.cc))

for (i in 1:length(alpha)) {
  for (j in 1:ncol(return.cc)) {
    s <- mean(abs(return.cc[,j])/rho.sign[j])
    for (k in 1:nrow(return.cc)) {
      error <- abs(return.cc[k,j])/rho.sign[j] - s
      SSE[i,j] <- error^2 + SSE[i,j]
      s.k <- alpha[i]*(abs(return.cc[k,j])/rho.sign[j]) + (1 - alpha[i])*s
      s <- s.k
    }
     alpha.opt.d[j] <- alpha[which.min(SSE[,j])]
  }
}
alpha.opt.d
```

```
## [1] 0.18 0.25 0.17 0.16
```

```
# find one-step ahead forward volatility forecast using optimal alpha
v.d <- c()
for (i in 1:ncol(return.cc)) {
  s <- mean(abs(return.cc[,i])/rho.sign[i])
  for (j in 1:nrow(return.cc)) {
    s.j <- alpha.opt.d[i]*(abs(return.cc[j,i])/rho.sign[i]) + (1 - alpha.opt.d[i])*s
    s <- s.j
  }
  v.d[i] <- s
}
v.d
```

```
## [1] 0.006727958 0.014341760 0.009881063 0.011558638
```

```
# find VaR forecast
VaR.d <- (-1)*v.d*tinv(p, df)
VaR.d
```

```
## [1] 0.03099412 0.06629923 0.04373438 0.05160141
```

```r
# summary statistics
summarystat <- data.frame(ss1, corr1, corr2, corr3, rho.sign, df, alpha.opt.e, alpha.opt.d, v.e, v.d, Va
summarystat
```

Numerical and Graphical Summary

```
##          mean.cc      sd.cc     min.cc    max.cc Excess.Kurtosis   Skewness
## BMO 0.0003041122 0.01431747 -0.1758104 0.1465208        29.53820 -1.0574354
## RBC 0.0005685011 0.02312980 -0.3237026 0.2246837        34.97794 -0.7034878
## TD  0.0003166799 0.01330011 -0.1498889 0.1489822        22.17936 -0.2174378
## BNS 0.0001316573 0.01378455 -0.1449471 0.1512738        23.56191 -0.3451518
##           corr1     corr2     corr3  rho.sign       df alpha.opt.e alpha.opt.d
## BMO -0.02468153 0.4742324 0.4222075 0.6298253 2.946908        0.25        0.18
## RBC -0.14958033 0.4339918 0.3916471 0.6281500 2.934449        0.30        0.25
## TD  -0.05523119 0.4474425 0.3956610 0.6480272 3.099599        0.21        0.17
## BNS -0.04149552 0.4835062 0.3606171 0.6442742 3.065250        0.25        0.16
##             v.e         v.d      VaR.e      VaR.d
## BMO 0.004441767 0.006727958 0.01033310 0.03099412
## RBC 0.012196756 0.014341760 0.02837390 0.06629923
## TD  0.007406026 0.009881063 0.01722899 0.04373438
## BNS 0.010039571 0.011558638 0.02335553 0.05160141
```

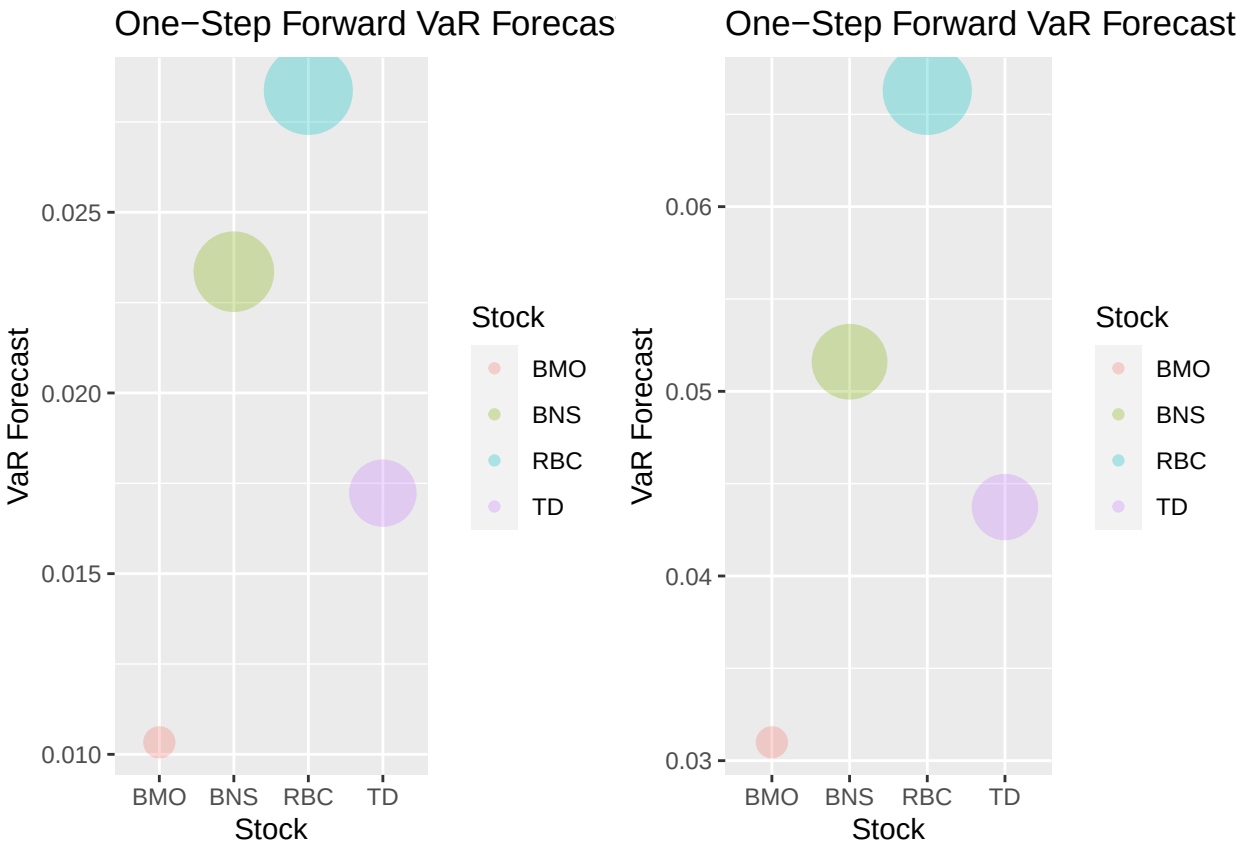
```r
# graphical summary for VaR forecasts
stock <- c("BMO","RBC","TD","BNS")
VaR.data <- data.frame(stock, VaR.e, VaR.d)

p.e <- VaR.data %>%
  ggplot(aes(x = stock, y = VaR.e, size = VaR.e, color = stock)) +
  labs(x = "Stock", y = "VaR Forecast", color = "Stock") +
  geom_point(alpha = 0.3) +
  scale_size(range = c(5, 15)) +
  guides(size = FALSE) +
  ggtitle("One-Step Forward VaR Forecast Using EWMA")
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
p.d <- VaR.data %>%
  ggplot(aes(x = stock, y = VaR.d, size = VaR.d, color = stock)) +
  labs(x = "Stock", y = "VaR Forecast", color = "Stock") +
  geom_point(alpha = 0.3) +
  scale_size(range = c(5, 15)) +
  guides(size = FALSE) +
  ggtitle("One-Step Forward VaR Forecast Using DDEWMA")

grid.arrange(p.e, p.d, ncol = 2)
```


One−Step Forward VaR Forecas
VaR Forecast
0.025
0.020
0.015
0.010
BMO
BNS
RBC
TD
Stock
Stock
BMO
BNS
RBC
TD
One−Step Forward VaR Forecast
VaR Forecast
0.06
0.05
0.04
0.03
BMO
BNS
RBC
TD
Stock
Stock
BMO
BNS
RBC
TD