STA457H1F Assignment 1

Ahmed Nawaz Amanullah (team lead), St.# 1001325773
Ling Long (team member), St.# 1002516618

February 24, 2019

Part A

Question 1

In this section, we will find the optimal double moving average (MA) trading rules for all 30 DJ constituents using monthly data. As hinted by the assignment, we will refer to the Fall 2018 assignment and implement the chronological steps necessary to retrieving these optimal values.

We will predict the direction of the trend of asset prices using a function of past asset prices F_t, which will be converted to buy and sell trading signals B_t, with buy corresponding to +1 and sell to -1. F_t will be based on a moving-average technical indicator, which can be expressed as a function of log returns:

$$F_t = \delta + \sum_{j=0}^{m-2} d_j X_{t-j}$$

where $X_t = ln(P_t/P_{t-1})$, while δ and d_j are defined by a given trading rule. For the assignment, we will assume $\delta = 0$.

The function is then, as mentioned earlier, converted to trading signals as follows:

$$\begin{cases} \text{"}Sell\text{"} \iff B_t = -1 \iff F_t < 0 \\ \text{"}Buy\text{"} \iff B_t = +1 \iff F_t > 0 \end{cases}$$

We will also be assessing the results and performance of trading decisions, and the returns at time t will be obtained using "ruled returns", which will be denoted as R_t. So if for a period $[t-1, t)$, a trader establishes a position at time $t-1$, represented by B_{t-1}, the "ruled returns" expression is as follows:

$$R_t = B_{t-1} X_t \iff \begin{cases} R_t = -X_t & B_{t-1} = -1 \\ R_t = +X_t & B_{t-1} = +1 \end{cases}$$

where $X_t = ln(P_t/P_{t-1})$ denote the logarithm return over this period (so we will be assuming no dividend payouts during the time period).

The realized returns will thus be determined using:

$$\tilde{R}_t = \sum_{D=1}^{n} R_{t+D}$$

where D represents the stochastic duration of the position lasting n days provided that:

$$\{D = n\} \iff \{B_{t-1} \neq B_t, B_t = ... = B_{t+n-1}, B_{t+n-1} \neq B_{t+n}\}$$

Step 1: Derive variance of predictor F_t

The variance of the predictor is derived as follows:

$$\sigma_F^2 = var(\sum_{i=0}^{m-2} d_i X_{t-i})$$

$$= cov(\sum_{i=0}^{m-2} d_i X_{t-i}, \sum_{i=0}^{m-2} d_i X_{t-i})$$

$$= d_0 cov(X_t, \sum_{i=0}^{m-2} d_i X_{t-i}) + d_1 cov(X_{t-1}, \sum_{i=0}^{m-2} d_i X_{t-i}) + ... + d_{m-2} cov(X_{t-m+2}, \sum_{i=0}^{m-2} d_i X_{t-i})$$

$$= d_0(d_0\gamma_0 + d_1\gamma_1 + ... + d_{m-2}\gamma_{m-2}) + d_1(d_0\gamma_1 + d_1\gamma_0 + d_2\gamma_1 + ... + d_{m-2}\gamma_{m-3}) + ... + d_{m-2}(d_0\gamma_{m-2} + d_1\gamma_{m-3} + ... + d_{m-2}\gamma_0)$$

$$= \gamma_0 \sum_{i=0}^{m-2} d_i^2 + 2 \sum_{i=1}^{m-2} \sum_{j=0}^{m-2-i} d_j d_{j+i} \gamma_i$$

$$= \gamma_0 (\sum_{i=0}^{m-2} d_i^2 + 2 \sum_{i=1}^{m-2} \sum_{j=0}^{m-2-i} d_j d_{j+i} \rho_i)$$

$$= \gamma_0 (\sum_{i=0}^{m-2} d_i^2 + 2 \sum_{j=0}^{m-2} \sum_{i=j+1}^{m-3} d_j d_i \rho_{i-j})$$

Provided below is the R code for computing the variance of the predictor:

```
#Step 1: Get the variance of the predictor
# d: vector of dj coefficients (j = 0, ..., m-2)
# X: log returns
#Will use quadratic form...
varF <- function(d, X){
    #maximum lag
    M <- length(d)-1
    #get auto-covariance
    acfs <- acf(X, plot = F, type="covariance", lag.max=M)$acf
    #get toeplitz matrix of acfs
    #name used to refer to:
    #[[\gamma_0, ..., \gamma_{m-2}],
    # [\gamma_1, \gamma_0, ..., \gamma_{m-3}], ...
    # [\gamma_{m-2}, ..., \gamma_0]]
    Gamma <- toeplitz(as.vector(acfs))
    #quadratic form
    varF <- d%*%Gamma%*%as.vector(d)
    varF
}
```

Note that the implementation above uses the quadratic form of the expression:

$$\sigma_F^2 = \begin{bmatrix} d_0 & ... & d_{m-2} \end{bmatrix} \begin{bmatrix} \gamma_0 & ... & \gamma_{m-2} \\ ... & ... & ... \\ \gamma_{m-2} & ... & \gamma_0 \end{bmatrix} \begin{bmatrix} d_0 \\ ... \\ d_{m-2} \end{bmatrix}$$

Step 2: Derive expectation of predictor F_t

The expectation of the predictor is derived as follows:

$$\begin{aligned}
\mu_F &= E(\sum_{i=0}^{m-2} d_i X_{t-i}) \\
&= \sum_{i=0}^{m-2} d_i E(X_{t-i}) \\
&= \mu_X \sum_{i=0}^{m-2} d_i
\end{aligned}$$

Provided below is the R code for computing the expectation of the predictor:

```
#Step 2: Get the expectation of the predictor
# d: vector of dj coefficients (j = 0, ..., m-2)
# X: log returns
muF <- function(d, X){
    muF <- mean(X)*sum(d)
    muF
}
```

Step 3: Computing the expected ruled returns

Under the assumption that X_t follows a stationary Gaussian process, the expected ruled returns has been provided as follows:

$$E(R_t) = \sqrt{\frac{2}{\pi}} \sigma_X corr(X_t, F_{t-1}) exp \left\{ -\frac{\mu_F^2}{2\sigma_F^2} \right\} + \mu_X (1 - 2\phi \left[-\frac{\mu_F}{\sigma_F} \right])$$

Most of the expressions needed for this computation has been derived earlier. We just need the expression:

$$\begin{aligned}
corr(X_t, F_{t-1}) &= \frac{cov(X_t, F_{t-1})}{\sqrt{\gamma_0 var(F_t)}} \\
&= \frac{(d_0\gamma_1 + d_1\gamma_2 + ... + d_{m-2}\gamma_{m-1})}{\sqrt{\gamma_0 var(F_t)}} \\
&= \frac{\gamma_0 \sum_{i=0}^{m-2} d_i \rho_{i+1}}{\sqrt{\gamma_0 var(F_t)}}
\end{aligned}$$

The R code for implementing this computation is as follows:

```
#Step 3 Intermediate: correlation between X_t and F_{t-1}
#will use a quadratic form
# d: vector of dj coefficients (j = 0, ..., m-2)
# X: log returns

corXF <- function(d, X){
    Mp <- length(d)
    acfs <- acf(X, plot = F, type = "covariance", lag.max = Mp)$acf
```

```
    corXF <- sum(d*acfs[-1])/sqrt(acfs[1]*varF(d, X))
    corXF
}
```

Provided below is the R code for the computation of the double MA co-efficients and the expected ruled returns:

```
#Step 3: Expected ruled returns

#next is the double MA co-efficients
d <- function(m, r){
    d <- c((m-r)*((0:(r-1))+1), r*(m-(r: (m-1))-1))
    d
}

# retX: log asset return
# m: long-term MA
# r: short-term MA

ruleReturn <- function(retX, m, r){
    #returns varianced
    vX <- sd(retX)
    #returns mean
    mX <- mean(retX)
    #predictor mean
    mF <- muF(d(m, r), retX)
    #predictor sd
    vF <- sqrt(varF(d(m, r), retX))
    #lag 1 correlation (rho) between X_t and F_{t-1}
    rXF <- corXF(d(m, r), retX)
    #expected return
    ER <- sqrt(2/pi)*vX*rXF*exp(-mF*mF/(vF*vF)) +
    mX*(1-2*pnorm(-mF/vF))
    #return results
    list("ER" = ER, "VF"= vF, "muF"=mF, "corXF"=rXF)
}
```

Step 4: Downloading the monthly data of the constituents

The data for the DJ constituents were downloaded from Yahoo Finance in csv files and stored in a folder. The R code for retrieving this data from the csv files is as follows:

```
#This function is to download data from folder...
#idea: return three arrays - one containing the stock tickers...
#   another the adjusted close (using hint from prev assn)
#   and finally one containing the log returns

# dataDir: directory containing csv files containing data
# currDir: current working directory (to change back to after
#   loading data)
getAdC <- function(dataDir, currDir){
    #switch to directory containing price data
```

```r
    setwd(dataDir)
    #get list of files in directory
    priceFiles <- list.files(dataDir, pattern="*.csv",
    full.names = F, recursive = F)
    #to hold tickers
    tickers <- c()
    #to hold prices
    prices <- c()
    #to hold log returns
    logrets <- c()
    #iterate through list above
    for (file in seq(1, length(priceFiles))){
        tickers <- c(tickers, substr(priceFiles[file],
        1,nchar(priceFiles[file])-4))
        #use rbind for the adjusted close data...
        #first read the file
        stockdat <- read.csv(file = priceFiles[file],
         header = T, sep=",")
        #then get adjusted close prices from file
        price <- stockdat[1:dim(stockdat)[1], 6]
        prices <- rbind(prices, price)
        #also the log returns
        logrets <- rbind(logrets,
        log(price[2:length(price)]/price[1:(length(price)-1)]))
    }
    #return to original directory
    setwd(currDir)

    #return tickers, prices and log returns
    list("tickers"=tickers, "prices"=prices, "logrets" = logrets)
}
```

Step 5: Writing R function to find optimal monthly double MA trading rules

Provided below are the R functions utilized to find the optimal double MA trading rules using monthly data:

```r
#Function to choose the optimal daily and monthly MA trading rules
#   (i.e. that maximize expected rule returns)
#will test function against the one in prev assn after
#   implementation...

#first optimal monthly
#   retX: vector of log returns
monthlyoptimal_dma <- function(retX){
    #to hold optimal m and r
    optimal_m <- 2
    optimal_r <- 1
    #get ruleReturn for this setting
    currER <- ruleReturn(retX, optimal_m, optimal_r)$ER
    #iterate up to 12 max as we are doing monthly
    #loop through r
    for (i in seq(1, 11)){
        for (j in seq(i+1, 12)){
```

```
            ERij <- ruleReturn(retX, j, i)$ER
            if (ERij > currER){
                optimal_m <- j
                optimal_r <- i
                currER <- ERij
            }
        }
    }
    #return optimal double MA trading rules
    list("monthlyoptimal_m"=optimal_m, "monthlyoptimal_r"=optimal_r)
}

#functions that will implement the above for all constituents...

#   retsX: the log returns of the constituents

monthlyoptimals_dma <- function(retsX){
    #to hold the optimal trading rules
    m <- c()
    r <- c()
    #amount of tickers to go through
    numstocks <- dim(retsX)[1]
    #number of periods/ months
    months <- dim(retsX)[2]
    for (i in seq(1, numstocks)){
        optimals <- monthlyoptimal_dma(retsX[i, 1:months])
        #add optimals to list
        m <- c(m, optimals$monthlyoptimal_m)
        r <- c(r, optimals$monthlyoptimal_r)
    }

    #return optimal rules
    list("monthlyoptimals_m"=m, "monthlyoptimals_r"=r)
}

#the version for the getting the optimal rules for each window

#   retsX: the log returns of the constituents

windowsmonthlyoptimals_dma <- function(retX){
    #number of stock
    num_stocks <- dim(retX)[1]
    #the number of periods
    months <- dim(retX)[2]
    #hold optimal m's and r's
    m <- matrix(,nrow = num_stocks, ncol = 1)
    r <- matrix(,nrow = num_stocks, ncol = 1)
    for (i in seq(1, months - 60, 12)){
        #get optimal dma within 60 month window for all stocks
        optimal_dma <- monthlyoptimals_dma(
        retX[1:num_stocks, i:(i+60-1)])
        if(i == 1){
            m[1:num_stocks, 1] <- optimal_dma$monthlyoptimals_m
```

```
            r[1:num_stocks, 1] <- optimal_dma$monthlyoptimals_r
        }else{
            m <- cbind(m, optimal_dma$monthlyoptimals_m)
            r <- cbind(r, optimal_dma$monthlyoptimals_r)
        }
    }

    list("m"=m, "r"=r)
}
```

Disclosed below are the optimal trading rules found for the constituents, using all the log return data:

```
## Warning in rbind(prices, price): number of columns of result is not a
## multiple of vector length (arg 2)

## Warning in rbind(logrets, log(price[2:length(price)]/price[1:(length(price)
## - : number of columns of result is not a multiple of vector length (arg 2)

## Warning: package 'knitr' was built under R version 3.2.5
```

Table 1: DJ double MA optimal trading rules

tickers	m	r
AAPL	3	2
AXP	2	1
BA	9	8
CAT	5	4
CSCO	7	6
CVX	8	6
DIS	5	4
DWDP	2	1
GS	2	1
HD	11	10
IBM	12	11
INTC	4	3
JNJ	9	8
JPM	12	11
KO	7	6
MCD	7	1
MMM	9	8
MRK	12	11
MSFT	9	8
NKE	10	7
PFE	10	8
PG	2	1
TRV	9	8
UNH	9	8
UTX	2	1
V	12	2
VZ	9	8
WBA	11	10
WMT	9	8
XOM	12	11

The optimal trading rules using the 60 months rolling window are provided below as well:

Table 2: Optimal m's for the rolling windows

tickers	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
AAPL	6	4	5	12	12	3	3	3	2	12	12	9	9	6
AXP	2	2	10	2	2	2	2	2	5	12	12	12	10	10
BA	3	2	7	12	12	6	9	5	5	12	12	12	7	10
CAT	6	10	10	12	2	5	5	5	5	2	4	12	3	7
CSCO	7	7	9	12	12	2	4	5	4	7	7	6	6	12
CVX	10	8	8	12	12	8	7	8	8	12	11	11	9	9
DIS	12	4	4	5	5	5	5	2	5	12	12	12	12	6
DWDP	8	9	9	4	2	2	2	2	2	2	4	4	11	11
GS	12	9	4	12	2	2	2	2	4	4	4	12	2	5
HD	5	3	3	3	10	9	9	9	12	12	12	12	12	12
IBM	6	12	6	10	3	3	3	2	5	5	7	12	6	3
INTC	4	7	5	8	2	2	3	2	2	10	10	7	2	12
JNJ	9	3	8	5	2	5	5	5	5	11	11	10	12	12
JPM	6	8	9	2	7	2	2	2	2	2	12	12	12	12
KO	3	3	3	7	7	7	7	5	5	12	12	12	12	7
MCD	4	4	3	12	12	12	12	12	6	7	12	12	4	4
MMM	7	9	10	10	8	2	9	5	4	12	12	12	12	12
MRK	7	12	12	12	3	9	9	9	9	9	11	10	4	4
MSFT	12	12	6	11	12	4	4	4	4	2	11	9	12	12
NKE	10	2	12	12	7	8	8	8	8	12	12	12	12	12
PFE	6	5	2	2	10	10	2	2	4	2	2	12	12	12
PG	2	4	12	12	12	3	3	2	2	3	12	7	7	7
TRV	6	2	3	2	2	6	9	9	9	12	12	12	12	12
UNH	12	12	12	12	9	9	9	9	9	12	12	12	12	12
UTX	2	2	7	12	6	6	5	5	5	12	12	3	3	3
V	5	12	12	12	12	12	12	11	5	5	5	5	12	12
VZ	9	9	9	11	9	9	9	5	5	10	10	10	12	12
WBA	5	3	3	10	7	6	6	6	6	11	11	11	11	11
WMT	9	12	12	12	6	6	5	5	9	10	10	10	3	3
XOM	12	11	8	12	8	8	12	3	2	2	3	11	7	7

Table 3: Optimal r's for the rolling windows

tickers	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
AAPL	5	3	1	10	11	1	1	1	1	1	1	1	1	5
AXP	1	1	9	1	1	1	1	1	4	11	9	11	9	9
BA	1	1	6	10	4	4	8	4	4	10	6	1	6	5
CAT	5	9	9	2	1	4	4	4	4	1	3	11	2	4
CSCO	6	6	8	11	11	1	3	3	3	5	6	5	5	10
CVX	6	7	7	6	11	7	6	4	7	11	10	10	8	8
DIS	11	3	3	4	4	4	4	1	4	11	11	11	8	5
DWDP	7	8	8	3	1	1	1	1	1	1	3	3	9	8
GS	11	8	3	6	1	1	1	1	1	3	3	11	1	4
HD	4	2	2	2	9	8	8	8	11	9	10	2	8	9
IBM	5	11	5	9	1	1	1	1	1	4	6	11	5	2
INTC	3	6	3	2	1	1	1	1	1	9	9	6	1	6
JNJ	8	2	7	4	1	4	4	4	4	8	7	8	3	3

tickers	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
JPM	5	7	8	1	6	1	1	1	1	1	11	11	6	7
KO	1	2	2	6	6	6	6	4	4	10	8	11	11	6
MCD	1	1	1	10	7	6	5	5	5	6	1	1	3	3
MMM	5	8	9	6	7	1	8	3	3	8	11	11	5	5
MRK	6	11	11	11	2	8	7	8	8	8	9	9	3	3
MSFT	10	11	5	10	11	3	3	3	3	1	10	8	7	6
NKE	8	1	1	6	6	6	7	7	7	6	7	9	1	11
PFE	5	4	1	1	9	9	1	1	3	1	1	9	9	9
PG	1	3	6	7	11	2	1	1	1	2	6	6	6	6
TRV	5	1	1	1	1	5	8	8	8	11	6	6	7	7
UNH	2	9	1	1	8	8	8	8	8	7	2	2	3	4
UTX	1	1	6	6	5	4	4	4	4	11	11	2	2	2
V	4	7	8	8	9	9	6	10	4	4	4	4	4	9
VZ	8	8	8	10	8	7	8	4	4	9	9	9	11	11
WBA	4	2	2	9	6	5	5	5	5	10	10	10	10	10
WMT	8	11	11	11	5	4	4	4	8	9	9	8	2	2
XOM	11	10	7	6	7	7	11	1	1	1	2	10	6	6

Question 2

In this section, we will construct the equally weighted (EW) and risk-parity (RP) weighted portfolios using the 30 DJ constituents and summarize the performance.

As per general instruction, performance will be based on a 60-month rolling window and the portfolio will be rebalanced monthly. The parameters (σ) will be calibrated/ estimated at the end of each year.

The following assumptions are going to be made:

1. Will use the 60-month window to get optimal trading rules.
2. Will use these optimal settings to predict the signals for next month
3. For risk parity, will use the last 12 months of window to calibrate the asset volatilities (see section B) as brought up earlier.
4. Use predicted signal and portfolio weights to get rule return of portfolio (see hint).
5. Check performance of next 12 months using predicted signals. Since we recalibrate parameters annually (i.e. at end of each year), we will re-calculate the ex-ante volatilities every 12 months.
6. The 60-month window moves 12 months ahead and we repeat (1-5). Process terminates when we reach the end of our data.

Please find the implementation of these steps for the equally weighted portfolio as follows:

```
#first function to compute the predictor

#   d: d is d(m ,r) - presumably the optimal ones found
#   retX: the log returns
#   t: get signal at time t

f <- function(d, retX, t){
    M <- length(d) - 1
```

```
    if (t >= M){
        output <- sum(d*rev(retX[(t-M):t]))
        output
    }
    else {print('t is smaller than M')}
}


#Question 2: summarizing performance of the equally weighted
#   portfolio
#   retX: 30X(periods) dim matrix of monthly log returns

ew_performance <- function(retX){
    #number of stocks provided
    num_stocks <- dim(retX)[1]
    #number of periods of data provided
    months <- dim(retX)[2]
    #to hold the ruled returns
    re <- c()
    #to hold the cumulative return up to time t
    cre <- c()
    #to hold expected return at time t
    ere <- c()
    #to hold the ex-ante volatility at time t
    ea_sd <- c()
    #to hold sharpe ratio at time t
    sharpes <- c()
    #only go up to as long as window can be fit
    for (i in seq(1, months - 60, 12)){
        #get optimal dma within 60 month window for all stocks
        optimal_dma <- monthlyoptimals_dma(
        retX[1:num_stocks, i:(i+60-1)])
        m <- optimal_dma$monthlyoptimals_m
        r <- optimal_dma$monthlyoptimals_r
        #need to get the ex-ante volatility every 12 months
        ea_sd_i <- ea_volatility(
        retX[1:num_stocks,1:(i+60-1)])
        #Ok, now we use the optimal trading rules to find the
        #   ruled return for each stock
        #to hold portfolio weights
        #this is an ew portfolio so all weights are the same
        weights <- rep(1/num_stocks, num_stocks)

        #this is the portfolio variance
        var_t <- 0
        for (s in seq(1, num_stocks)){
            var_t <- var_t + (weights[s]^2)*ea_sd_i[s]^2
        }

        ea_sd <- c(ea_sd, sqrt(var_t))

        #going back to old method
        for (t in seq(i + 60, i + 72 -1)){
```

```r
            if (t <= months){
                re_t <- 0
                for (s in seq(1, num_stocks)){
                    s_d <- d(m[s], r[s])
                    re_t <- re_t + weights[s]*
                    sign(f(s_d, retX[s, 1:months], t-1))*retX[s,t]
                }
                #now add the portfolio ruled return
                re <- c(re, re_t)
                if (length(re) > 1){
                    cre <- c(cre, cre[length(cre)] + re_t)
                }else{
                    cre <- c(cre, re_t)
                }
                #expected (annual) return at time t
                ere <- c(ere, (12*cre[length(cre)]/length(cre)))
                #as well the sharpe
                sharpe_t <- (ere[length(ere)]-0.02)/
                ea_sd[length(ea_sd)]
                sharpes <- c(sharpes, sharpe_t)
                #note that since this is a EW portfolio, there is
                #   no rebalancing step as weights remain the same
            }
        }


    }

    #return results
    list("return"=ere[length(ere)],
    "volatility"=ea_sd[length(ea_sd)],
    "sharpe"=sharpes[length(sharpes)], "cumul_return"=cre,
    "e_return"=ere, "ea_volatilities"=ea_sd, "sharpes"=sharpes)
}

#risk parity function
rp_performance <- function(retX){
    #number of stocks provided
    num_stocks <- dim(retX)[1]
    #number of periods of data provided
    months <- dim(retX)[2]
    #to hold the ruled returns
    re <- c()
    #to hold the cumulative return up to time t
    cre <- c()
    #to hold expected return at time t
    ere <- c()
    #to hold the ex-ante volatility at time t
    ea_sd <- c()
    #to hold sharpe ratio at time t
    sharpes <- c()
    #only go up to as long as window can be fit
    for (i in seq(1, months - 60, 12)){
```

```
    #get optimal dma within 60 month window for all stocks
    optimal_dma <- monthlyoptimals_dma(
    retX[1:num_stocks, i:(i+60-1)])
    m <- optimal_dma$monthlyoptimals_m
    r <- optimal_dma$monthlyoptimals_r
    #need to get the ex-ante volatility every 12 months
    ea_sd_i <- ea_volatility(
    retX[1:num_stocks,1:(i+60-1)])
    #Ok, now we use the optimal trading rules to find the
    #   ruled return for each stock
    #to hold portfolio weights
    #this is an ew portfolio so all weights are the same
    temp <- 1/ea_sd_i
    weights <- temp/sum(temp)

    #this is the portfolio variance
    var_t <- 0
    for (s in seq(1, num_stocks)){
        var_t <- var_t + (weights[s]^2)*ea_sd_i[s]^2
    }

    ea_sd <- c(ea_sd, sqrt(var_t))

    #going back to old method
    for (t in seq(i + 60, i + 72 -1)){
        if (t <= months){
            re_t <- 0
            for (s in seq(1, num_stocks)){
                s_d <- d(m[s], r[s])
                re_t <- re_t + weights[s]*
                sign(f(s_d, retX[s, 1:months], t-1))*retX[s,t]
            }
            #now add the portfolio ruled return
            re <- c(re, re_t)
            if (length(re) > 1){
                cre <- c(cre, cre[length(cre)] + re_t)
            }else{
                cre <- c(cre, re_t)
            }
            #expected (annual) return at time t
            ere <- c(ere, (12*cre[length(cre)]/length(cre)))
            #as well the sharpe
            sharpe_t <- (ere[length(ere)]-0.02)/
            ea_sd[length(ea_sd)]
            sharpes <- c(sharpes, sharpe_t)
            #note that since this is a EW portfolio, there is
            #   no rebalancing step as weights remain the same
        }
    }


}
```

```
    #return results
    list("return"=ere[length(ere)],
    "volatility"=ea_sd[length(ea_sd)],
    "sharpe"=sharpes[length(sharpes)], "cumul_return"=cre,
    "e_return"=ere, "ea_volatilities"=ea_sd, "sharpes"=sharpes)
}
```

The performance results obtained implementing the function above is as follows (note that in the plots presented, the black line corresponds to the equally weighted portfolio and the blue to the risk parity portfolio):

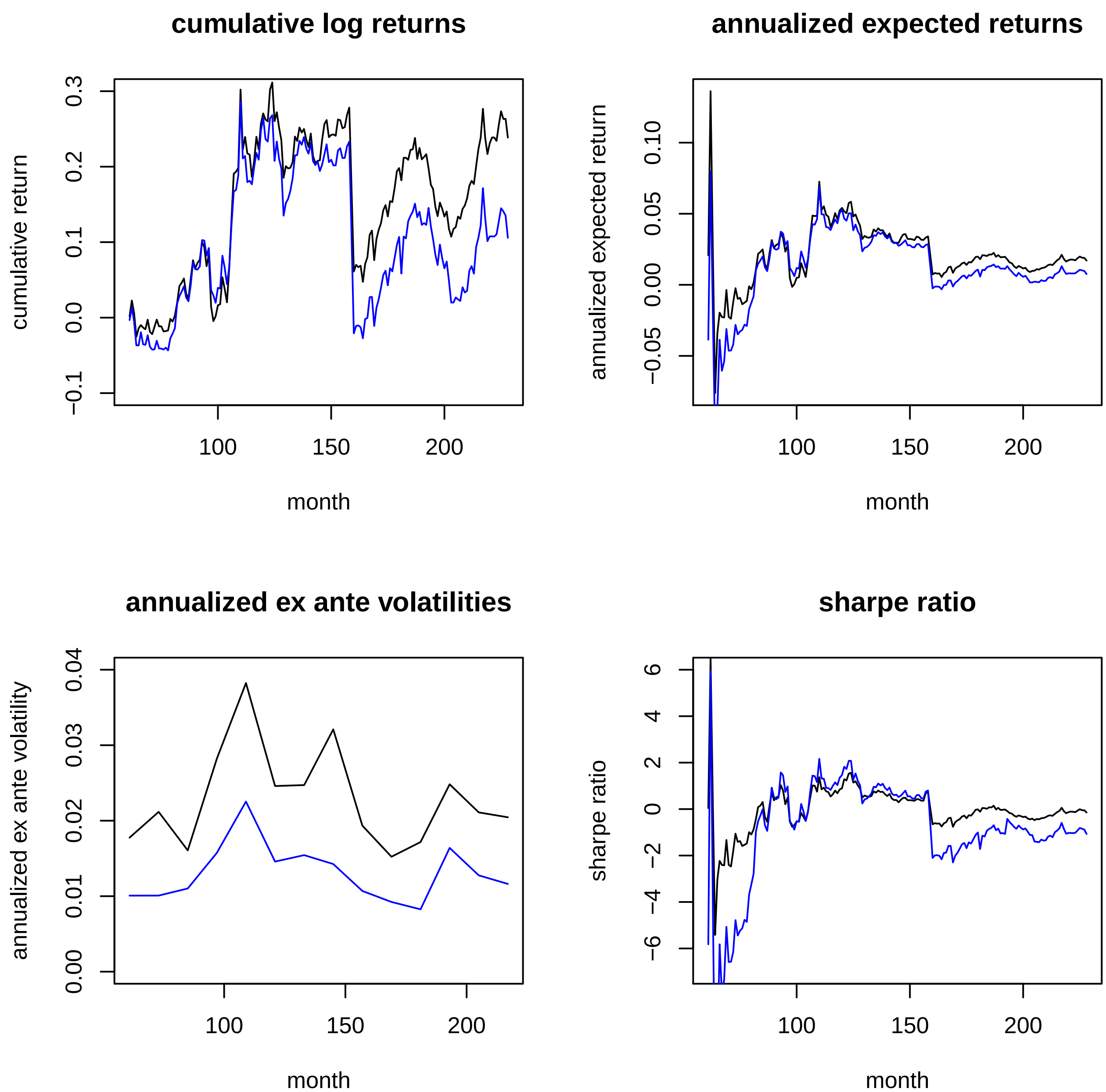

Table 4: EW performance metrics

Annualized expected return	Annualized volatility	Annualized sharpe ratio
0.017	0.0205	-0.1448

Table 5: RP performance metrics

Annualized expected return	Annualized volatility	Annualized sharpe ratio
0.0076	0.0116	-1.0707

Part B

Question 1

In this section, we will cover the computation of the ex-ante volatility estimate σ_t for the DJ constituents. The formula used is:

$$\sigma_{s,t}^2 = 12\sum_{i=0}^{11}(1-\delta)\delta^i(r_{s,t-1-i} - \bar{r}_{s,t})^2$$

Where the weights $\delta^i(1-\delta)$ add up to one, and $\bar{r}_{s,t}$ is the exponentially weighted average return computed similarly:

$$\bar{r}_{s,t} = \sum_{i=0}^{11}(1-\delta)\delta^i r_{s,t-1-i}$$

For this assignment, as instructed, we will use $\delta = 0.2$.

The code for implementing this computation in R is as follows:

```
ea_volatility <- function(retX){
    #need to solve for delta
    #according to paper delta has been solved for as
    delta <- 0.2
    #number of stocks
    num_stocks <- dim(retX)[1]
    #number of periods provided
    t <- dim(retX)[2]
    #vector to hold the ex ante volatilities
    ea_sd <- c()
    #need to iterate over the stocks
    for (s in seq(1, num_stocks)){
        #compute the exponentially weighted returns
        r_bar <- sum((1-delta)*(delta^c(0:11))*
        rev(retX[s, (t-12):(t-1)]))
        #next the ex-ante volatility (annualized)
        #first the variance
        var <- 12*sum((1-delta)*(delta^c(0:11))*
        (rev(retX[s, (t-12):(t-1)])-r_bar)^2)
        #then add the ex-ante volatility of stock s to list
        ea_sd <- c(ea_sd, sqrt(var))
    }

    #return volatilities
    ea_sd
}
```

Question 2

In this section, we will consider the predictive regression that regresses the (excess) return in month t on its return lagged by h months:

$$\frac{r_{s,t}}{\sigma_{s,t-1}} = \alpha + \beta_h \cdot \frac{r_{s,t-h}}{\sigma_{s,t-h}} + \epsilon_{s,t}$$

and

$$\frac{r_{s,t}}{\sigma_{s,t-1}} = a + b_h \cdot sign(r_{s,t-h}) + \xi_{s,t}$$

where $r_{s,t}$ denotes the s-th stock in the DJ constituents and the returns are scaled by the ex-ante volatilities, i.e. $\sigma_{s,t-1}$ as calculated above in question 1 of this section. We want to determine the optimal value of h for all 30 DJ constituents. For simplicity, as specified in the assignment sheet, we only use the latter formula as specified above.

We compute $\sigma_{s,t}$ for all possible combinations s, t before we loop through potential h values, in order to reduce computational time. Since we now use the past 12 months of data to calculate the variance, we actually range through 12x18 months instead of 12x19.

First, we use the $ea_volatility$ function defined in part 1 to construct our matrix of ex-ante volatilities, since we need the volatility one step backwards in time at any point for our regression model. We implement this by simply looping over the possible months, and computing the volatility at each point and putting it into a matrix.

```
ea_volatility_monthly <- function(retX){
  months <- dim(retX)[2]
  numStocks <- 30
  res <- c()

  for (t in c(13:228)){
    value <- ea_volatility(retX[1:numStocks, 1:t])
    res <- cbind(res, c(value))
  }

  return(res)
}
```

Here, we use our function from above to pre-compute volatilities. In order to build the regression, we simply build datasets for every possible time lag h. For example, for h = 1, we will build a dataset with every possible value of t such that t-h is above 0, so that we can build a regression model using $r_{s,t}$ as the dependant variable, and our $sign(r_{s,t-h})$ as our independant variable. In order to judge the linear models for each value of H, we use the R-squared value, with higher being better. We note that building the dataset in this way tends to skew the optimal h-values in favor of larger h. If h is larger, the set of data points which we can build our model with is restricted, and it is possible that the linear model built using a smaller set of data points is very linear (having an R-squared value of close to 1), and as such will be chosen as our optimal H.

```
predictive_regression <- function(retX){
  num_stocks <- 30
  #number of periods of data provided

  optimal_h <- c()
  volatilities <- ea_volatility_monthly(mlogrets)
  months <- dim(volatilities)[2]
```

```
  for (s in c(1:num_stocks)){
     max_r <- 0
     best_h <- c(50)
     for (h in seq(1,228-12)){
         y <- c()
         x <- c()
         for (t in seq(h, months)){
            y <- c(y, (retX[s, t+1]/volatilities[s, t]))
            x <- c(x, sign(retX[s, t-h+1]))
         }

         relation <- data.frame("x" = x, "y" = y)

         linearMode <- lm(y ~ x, data = relation)
         rsquared <- summary(linearMode)$r.squared

         if (rsquared > max_r){
           max_r = rsquared
           best_h <- c(h)
         }
     }
     optimal_h <- c(optimal_h, best_h)
  }
  return(optimal_h)
}
```

Table 6: Optimal H values for each of the 30 DJ constituents

tickers	Optimal H Value
AAPL	206
AXP	214
BA	215
CAT	214
CSCO	208
CVX	214
DIS	215
DWDP	214
GS	215
HD	215
IBM	215
INTC	209
JNJ	210
JPM	215
KO	213
MCD	213
MMM	213
MRK	215
MSFT	214
NKE	209
PFE	215
PG	211
TRV	214
UNH	201

tickers	Optimal H Value
UTX	214
V	214
VZ	215
WBA	211
WMT	213
XOM	215

Question 3

Similar to Part A question 2, we will construct a time series momentum (TSMOM) trading strategy weighted portfolio using the 30 DJ constituents and summarize the performance.

Again, performance will be based on a 60-month rolling window and the portfolio will be rebalanced monthly. The parameters (σ) will be calibrated/ estimated at the end of each year. In order to display the performance, we show the cumulative log returns, annualized expected returns, annualized ex ante volatilities, and sharpe ratio.

The following assumptions are going to be made:

1. The 60-month window rolling window will be applied to the model.
2. The last 12 months of data will be used to calculate our signal to predict the next month..
3. The last 12 months will be used calibrate the asset volatilities as described in Question 1 of Part B.
4. We will use predicted signal and portfolio weights to get rule return of portfolio.
5. Check performance of next month using predicted signal. Since we recalibrate parameters annually (i.e. at end of each year), we will re-calculate the ex-ante volatilities every 12 months.
6. The 60-month window moves 1 months ahead (as we rebalance monthly as per general instructions) and we repeat (1-5). Process terminates when we reach the end of our data.

Here, we have that the returns are determined as:

$$r^{TSMOM}_{t,t+1} = \frac{1}{30}\sum_{s=1}^{30} sign(r_{s,t-h_s:t}) \cdot \frac{40\%}{\sigma_{s,t}} r_{s,t:t+1}$$

where we have that:

$$B_{s,t} = sign(r_{s,t-h_s:t}) \cdot \frac{40\%}{\sigma_{s,t}}$$

is our position for the s-th constituent at time t and $r_{h_s:t-h_s:t}$ denotes the h_s-month lagged returns observed at time t. As per the hint given, we take $h_s = 12$ for all 30 constituents.

```
tsmom_performance <- function(retX){
  #number of stocks provided
  num_stocks <- dim(retX)[1]
  #number of periods of data provided
  months <- dim(retX)[2]
  #to hold the ruled returns
  re <- c()
  #to hold the cumulative return up to time t
  cre <- c()
```

```r
#to hold expected return at time t
ere <- c()
#to hold the ex-ante volatility at time t
ea_sd <- c()
#to hold sharpe ratio at time t
sharpes <- c()
#only go up to as long as window can be fit
for (i in seq(1, months - 60, 12)){
  ea_sd_i <- ea_volatility(
    retX[1:num_stocks,1:(i+60-1)])
  #Ok, now we use the optimal trading rules to find the
  #ruled return for each stock
  #to hold portfolio weights
  # weigh them as 0.4/volatility
  temp <- 0.4/ea_sd_i
  weights <- temp/sum(temp)

  #this is the portfolio variance
  var_t <- 0
  for (s in seq(1, num_stocks)){
    var_t <- var_t + (weights[s]^2)*ea_sd_i[s]^2
  }

  ea_sd <- c(ea_sd, sqrt(var_t))

  for (t in seq(i + 60, i + 72 -1)){
    if (t <= months){
      re_t <- 0
      for (s in seq(1, num_stocks)){
        re_t <- re_t +  weights[s]*sign(sum(retX[s,t-12:t]))*(0.4/ea_sd_i[s])*retX[s,t]
      }
      #now add the portfolio ruled return
      re <- c(re, re_t)
      if (length(re) > 1){
        cre <- c(cre, cre[length(cre)] + re_t)
      }else{
        cre <- c(cre, re_t)
      }
      #expected (annual) return at time t
      ere <- c(ere, (12*cre[length(cre)]/length(cre)))
      #as well the sharpe
      sharpe_t <- (ere[length(ere)]-0.02)/
        ea_sd[length(ea_sd)]
      sharpes <- c(sharpes, sharpe_t)
    }
  }
}

#return results
list("return"=ere[length(ere)],
     "volatility"=ea_sd[length(ea_sd)],
     "sharpe"=sharpes[length(sharpes)], "cumul_return"=cre,
     "e_return"=ere, "ea_volatilities"=ea_sd, "sharpes"=sharpes)
```

```
}
```

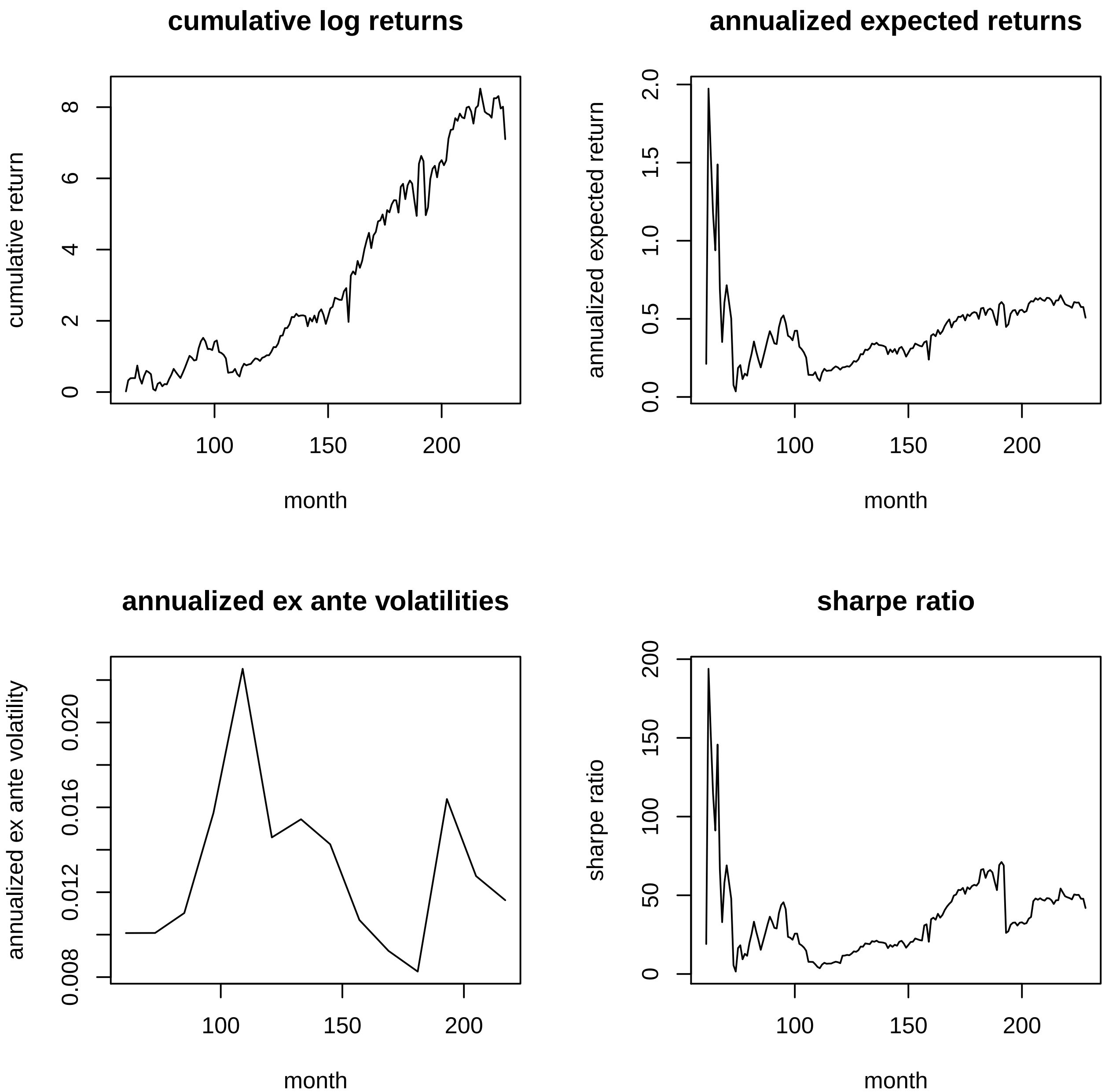

Table 7: EW performance metrics

Annualized expected return	Annualized volatility	Annualized sharpe ratio
0.0507	0.0116	4.191

Part C

Question 1

In this section, we will make it so that our position to the trading rule is determined by the magnitude of the signal. First, we will need to compute the expected h-period holding period return. We have been provided the technical indicator F_t:

$$F_t = \sum_{i=0}^{m-2} d_i r_{t-i}$$

As well as the h-period holding return expression:

$$R_{t:t+h} = \sum_{j=0}^{h-1} F_{t+j} r_{t+j+1} = \sum_{j=0}^{h-1} \left(\sum_{i=0}^{m-2} d_i r_{t+j-i} \right) r_{t+j+1}$$

The expected h-period holding period return is computed as follows:

$$\begin{aligned}
E(R_{t:t+h}) &= E\left(\sum_{j=0}^{h-1} \left(\sum_{i=0}^{m-2} d_i r_{t+j-i} \right) r_{t+j+1} \right) \\
&= E(r_{t+1}(d_0 r_t + d_1 r_{t-1} + ... + d_{m-2} r_{t-m+2}) + ... + r_{t+h}(d_0 r_{t+h-1} + ... + d_{m-2} r_{t+h-m+1})) \\
&= d_0(E(r_{t+1} r_t) + ... + E(r_{t+h} r_{t+h-1})) + ... + d_{m-2}(E(r_{t+1} r_{t-m+2}) + ... + E(r_{t+h} r_{t+h-m+1})) \\
&= d_0(h\gamma_1 + hE(r_t)^2) + ... + d_{m-2}(h\gamma_{m-1} + hE(r_t)^2) \\
&= h \sum_{i=0}^{m-2} d_i \gamma_{i+1} + hE(r_t)^2 \sum_{i=0}^{m-2} d_i \\
&= h \sum_{i=0}^{m-2} d_i (\gamma_{i+1} + E(r_t)^2)
\end{aligned}$$

The R-code for the computation of the expected h-period holding period return is as follows:

```r
#Function to compute the expected h-period holding period return

# retX: monthly log return
# m: long-term MA
# r: short-term MA
# h: holding period

e_hperiod_holdperiodreturn <- function(retX, m, r, h){
    #get auto-covariances
    acfs <- acf(retX, plot = F, type = "covariance", lag.max = m)$acf
    #get expected log return
    ER <- mean(retX)
    #get expected holding period return
    ehphpr <- h*sum(d(m,r)*(acfs[2:length(acfs)] + (ER^2)))
    #return value
    ehphpr
}
```

Question 2

In this section we will find the optimal double MA for all 30 DJ constituents that maximize the 12-period holding period return. Please find enclosed below the R-function that was written to implement this:

```r
#Ques 2: optimal double MA for 30 DJ constituents maximizing
#   12-period holding period return.

# retX: monthly log returns

monthlyoptimalEHR_dma <- function(retX){
    #to hold optimal m and r
    optimal_m <- 2
    optimal_r <- 1
    #get ruleReturn for this setting
    currEHR <- e_hperiod_holdperiodreturn(retX,
    optimal_m, optimal_r, 12)
    #iterate up to 12 max as we are doing monthly
    #loop through r
    for (i in seq(1, 11)){
        for (j in seq(i+1, 12)){
            EHRij <- e_hperiod_holdperiodreturn(retX,
            j, i, 12)
            if (EHRij > currEHR){
                optimal_m <- j
                optimal_r <- i
                currEHR <- EHRij
            }
        }
    }
    #return optimal double MA trading rules
    list("monthlyoptimal_m"=optimal_m, "monthlyoptimal_r"=optimal_r)
}

# retsX: monthly log returns

monthlyoptimalsEHR_dma <- function(retsX){
    #to hold the optimal trading rules
    m <- c()
    r <- c()
    #amount of tickers to go through
    numstocks <- dim(retsX)[1]
    #number of periods/ months
    months <- dim(retsX)[2]
    for (i in seq(1, numstocks)){
        optimals <- monthlyoptimalEHR_dma(retsX[i, 1:months])
        #add optimals to list
        m <- c(m, optimals$monthlyoptimal_m)
        r <- c(r, optimals$monthlyoptimal_r)
    }

    #return optimal rules
    list("monthlyoptimals_m"=m, "monthlyoptimals_r"=r)
}
```

```r
#the version for the getting the optimal rules for each window

# retX: monthly log returns

windowsmonthlyoptimalsEHR_dma <- function(retX){
    #number of stock
    num_stocks <- dim(retX)[1]
    #the number of periods
    months <- dim(retX)[2]
    #hold optimal m's and r's
    m <- matrix(,nrow = num_stocks, ncol = 1)
    r <- matrix(,nrow = num_stocks, ncol = 1)
    for (i in seq(1, months - 60, 12)){
        #get optimal dma within 60 month window for all stocks
        optimal_dma <- monthlyoptimalsEHR_dma(
        retX[1:num_stocks, i:(i+60-1)])
        if(i == 1){
            m[1:num_stocks, 1] <- optimal_dma$monthlyoptimals_m
            r[1:num_stocks, 1] <- optimal_dma$monthlyoptimals_r
        }else{
            m <- cbind(m, optimal_dma$monthlyoptimals_m)
            r <- cbind(r, optimal_dma$monthlyoptimals_r)
        }
    }

    list("m"=m, "r"=r)
}
```

Disclosed below are the trading rules for optimal expected 12-period holding period return for the constituents, using all the log return data:

Table 8: DJ double MA trading rules for optimal expected 12-period holding period return

tickers	m	r
AAPL	12	5
AXP	10	3
BA	12	6
CAT	10	4
CSCO	12	5
CVX	12	6
DIS	12	11
DWDP	5	3
GS	4	1
HD	12	7
IBM	12	11
INTC	8	3
JNJ	9	8
JPM	2	1
KO	7	6
MCD	12	5
MMM	10	8
MRK	12	6

tickers	m	r
MSFT	12	8
NKE	12	6
PFE	12	8
PG	6	1
TRV	2	1
UNH	12	7
UTX	6	4
V	12	4
VZ	12	7
WBA	11	10
WMT	9	8
XOM	12	6

The optimal trading rules using the 60 months rolling window are provided below as well:

Table 9: Optimal m's for the rolling windows

tickers	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
AAPL	12	12	12	12	12	12	12	12	12	12	12	12	12	12
AXP	10	11	12	12	12	9	9	9	9	12	12	12	12	12
BA	5	12	12	12	12	12	12	12	12	12	12	12	12	12
CAT	10	12	12	12	6	9	9	9	9	12	12	12	3	12
CSCO	12	9	12	12	12	12	12	5	6	11	12	7	7	12
CVX	12	12	12	12	12	12	12	10	8	12	12	12	12	12
DIS	12	12	5	12	7	10	12	6	6	12	12	12	12	12
DWDP	10	9	12	12	12	7	7	6	6	2	5	12	12	12
GS	12	12	12	12	12	12	9	9	6	12	4	12	12	12
HD	8	9	11	12	12	10	12	12	12	12	12	12	12	12
IBM	6	12	6	12	12	6	6	6	8	12	12	12	3	3
INTC	8	8	10	12	12	6	6	6	6	11	12	7	7	12
JNJ	9	9	9	8	7	6	5	5	5	12	12	12	12	12
JPM	9	9	9	12	12	2	2	2	2	12	12	12	12	12
KO	3	5	5	12	12	12	12	12	12	12	12	12	9	9
MCD	9	9	9	12	12	12	12	12	12	12	12	12	7	12
MMM	10	12	12	12	12	9	9	6	6	12	12	12	12	12
MRK	7	12	12	12	12	12	12	11	12	12	12	12	12	12
MSFT	12	12	6	11	12	12	4	4	7	9	11	12	12	12
NKE	10	9	12	12	12	12	12	12	11	12	12	12	12	12
PFE	10	12	11	11	12	12	2	6	6	5	12	12	12	12
PG	6	12	12	12	12	5	5	5	5	7	12	8	10	12
TRV	2	3	4	10	8	9	9	9	9	12	12	12	12	12
UNH	12	12	12	12	12	10	12	12	12	12	12	12	12	12
UTX	2	4	12	12	12	12	12	10	10	12	12	12	12	12
V	12	12	12	12	12	12	12	11	5	5	12	12	12	12
VZ	9	9	9	12	12	12	12	12	12	12	12	12	12	10
WBA	8	8	9	10	12	10	6	6	6	11	11	11	11	11
WMT	9	12	12	12	6	9	6	6	10	12	12	11	7	11
XOM	12	12	12	12	12	12	12	8	5	7	6	12	11	9

Table 10: Optimal r's for the rolling windows

tickers	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
AAPL	10	7	7	7	8	8	2	5	2	6	6	5	5	5
AXP	9	8	7	6	5	3	3	3	3	7	8	6	6	6
BA	1	11	6	6	6	4	4	4	4	7	6	6	5	6
CAT	4	8	5	5	3	4	4	4	4	11	11	11	2	5
CSCO	5	6	6	7	8	8	11	3	3	10	10	5	5	5
CVX	6	6	6	6	6	6	5	4	6	5	10	8	6	6
DIS	11	11	3	4	4	3	4	4	3	7	3	8	6	5
DWDP	7	8	8	3	5	1	1	1	1	1	3	10	5	5
GS	10	11	7	6	7	1	1	1	2	3	3	8	6	4
HD	4	1	2	8	8	8	8	8	8	8	5	6	6	6
IBM	5	11	3	9	11	1	2	2	2	6	4	9	2	2
INTC	3	6	3	3	9	2	2	1	3	9	6	5	6	6
JNJ	7	7	6	4	4	4	4	4	4	7	7	6	6	5
JPM	5	5	5	6	6	1	1	1	1	11	11	10	6	6
KO	1	2	2	6	4	4	4	4	4	4	8	8	8	5
MCD	3	2	2	7	7	6	6	6	5	6	6	5	3	5
MMM	5	8	8	6	6	5	8	3	3	8	7	8	5	5
MRK	6	11	11	9	8	7	6	7	7	8	7	7	9	6
MSFT	8	11	5	10	10	1	3	3	3	6	7	7	7	7
NKE	7	4	5	6	5	5	6	6	7	6	7	6	5	5
PFE	5	9	7	9	8	8	1	1	2	1	8	8	9	9
PG	3	3	6	7	11	4	1	4	4	6	6	6	6	6
TRV	1	1	1	6	5	5	5	8	8	6	6	6	7	8
UNH	6	6	6	6	5	8	8	7	7	7	7	7	7	6
UTX	1	3	5	6	5	4	4	4	4	4	9	11	4	4
V	3	7	6	6	5	5	6	10	4	3	4	4	6	6
VZ	7	8	8	8	7	7	7	4	4	8	6	8	9	8
WBA	4	2	6	6	6	5	5	5	5	10	10	10	10	10
WMT	8	11	11	11	4	4	4	3	8	8	7	7	2	4
XOM	9	7	6	6	7	6	6	1	1	1	2	10	6	6