Portfolio Optimization

Neha Awasthi

2022-10-30

Introduction

Portfolio optimization is the process of selecting the best portfolio, out of the set of all feasible portfolios for our desired outcome. We typically maximize return and minimizes financial risk during the consideration. Sharpe Ratio is a measure of calculating risk adjusted return. Higher the Sharpe Ratio, more is the excess returns over that of a risk free investment, relative to the increased risk of the investment.

In this project, we are going to use Monte-Carlo simulation method to optimize portfolios using R. We write a function that takes Tickers, beginning date, end date and risk-free rate as input and gives us a feasible set. We optimize it using weight simulation and find an efficient frontier and then try to find an optimal portfolio.

Let's load all the libraries needed first

```
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

Let's write an R function that a vector of TICKERs, a begin date, an end date, (annualized) risk-free rate as arguments and produce an list containing the vector of stock means, the covariance matrix and relevant information (weights, mean, sigma, SR) for each simulated portfolio.

```
myMeanVarPort <- function(ticks,begin_date,end_date,rf_rate){

  #Create a portfolio for the required tickers and given dates
  retout <- NULL
  retout <- xts(retout)

  for(i in 1:length(ticks)){
    prices = getSymbols(ticks[i], auto.assign = F)
    returns <- periodReturn(prices, period = "monthly",
                          type = "arithmetic")
    retout <- merge.xts(retout, returns)
  }

  colnames(retout) <- ticks
  dates <- paste(begin_date, end_date, sep='/')
  retout = retout[dates]

  # Calculate mean
  meanret <- colMeans(retout,na.rm = T)

  # Get Covariance matrix
  covar <- var(retout)

  meanret <- as.matrix(meanret)

  # Simulate weights for the portfolios for 100N portfolios
  set.seed(12)
  niter <- 100*length(ticks)
  randomnums <- data.frame(replicate(length(ticks), runif(niter, 1, 10)))

  wt_sim <- randomnums / rowSums(randomnums)

  ## initialize weight and Results matrices
  weight <- matrix(data = NA, nrow = length(ticks), ncol = 1)

  Results <- matrix(data = NA, nrow = niter, ncol = (length(ticks)+3))

  #run the simulations.

  # loop: each i is a portfolio
  for (i in 1:niter){
    # inner loop places weights into Results
    for (k in 1:length(ticks)) {
      Results[i,k] = weight[k,1] = wt_sim[i,k]
    }
    Results[i,length(ticks)+1] <- t(weight) %*% meanret                #portfolio mean
    Results[i,length(ticks)+2] <- sqrt(t(weight) %*% covar %*% weight) #portfolio sigma
    Results[i,length(ticks)+3] <- (Results[i,length(ticks)+1] - rf_rate)/Results[i,length(ticks)+2] #po
```

```
  }

  #Calculate Portfolio Mean return, risk and Sharpe ratio
  mretp <- t(weight) %*% meanret
  sretp <- sqrt(t(weight) %*% covar %*% weight)
  shrretp <- (mretp - rf_rate)/sretp

  # Store the output in the list containing the vector of stock means, the covariance matrix, weights, 
  out_lst <-list()
  out_lst[["Stock_means"]] <- meanret
  out_lst[["Covariance_Matrix"]] <- covar
  out_lst[["Portfolio_Weight"]] <- weight
  out_lst[["Portfolio_Mean"]] <- mretp
  out_lst[["Portfolio_Sigma"]] <- sretp
  out_lst[["Portfolio_Sharpe_ratio"]] <- shrretp
  out_lst[["Results"]] <- Results
  return(out_lst)
}
```

Show results for the given inputs

```
ticks <- c('GE', 'XOM', 'GBX', 'SBUX', 'PFE', 'HMC', 'NVDA')
begin_date <- 20140101
end_date <- 20171231
rf_rate <- 0.02

output <- myMeanVarPort(ticks,begin_date,end_date,rf_rate)
```

The vector of stock means

```
output[["Stock_means"]]
```

```
##             [,1]
## GE   -0.008371841
## XOM  -0.003140133
## GBX   0.015364882
## SBUX  0.009024095
## PFE   0.004415191
## HMC  -0.002464943
## NVDA  0.058165669
```

Covariance Matrix:

```
output[["Covariance_Matrix"]]
```

```
##                GE          XOM          GBX         SBUX          PFE
## GE   0.0029425540 0.0010346975 1.211210e-03 8.072761e-04 0.0006465268
## XOM  0.0010346975 0.0016851744 1.625325e-03 2.360760e-04 0.0005344605
## GBX  0.0012112097 0.0016253253 1.036163e-02 6.399308e-05 0.0015767495
## SBUX 0.0008072761 0.0002360760 6.399308e-05 2.115326e-03 0.0005236355
## PFE  0.0006465268 0.0005344605 1.576749e-03 5.236355e-04 0.0018874080
## HMC  0.0006216792 0.0002043222 2.127414e-03 6.802793e-04 0.0008354260
## NVDA 0.0013543229 0.0010916138 3.182648e-03 3.671076e-04 0.0003674847
##               HMC         NVDA
## GE   0.0006216792 0.0013543229
```

```
## XOM  0.0002043222 0.0010916138
## GBX  0.0021274140 0.0031826485
## SBUX 0.0006802793 0.0003671076
## PFE  0.0008354260 0.0003674847
## HMC  0.0031606549 0.0011638388
## NVDA 0.0011638388 0.0111256402
```

Weight:

```
output[["Portfolio_Weight"]]
```

```
##            [,1]
## [1,] 0.13229426
## [2,] 0.23842279
## [3,] 0.04149393
## [4,] 0.13445616
## [5,] 0.21049501
## [6,] 0.18606403
## [7,] 0.05677381
```

Portfolio Mean:

```
output[["Portfolio_Mean"]]
```

```
##             [,1]
## [1,] 0.003767694
```

Portfolio Sigma:

```
output[["Portfolio_Sigma"]]
```

```
##            [,1]
## [1,] 0.03192729
```

Portfolio Sharpe Ratio:

```
output[["Portfolio_Sharpe_ratio"]]
```

```
##            [,1]
## [1,] -0.5084148
```

Let's check the data type of output of our function

```
class(output)
```

```
## [1] "list"
```

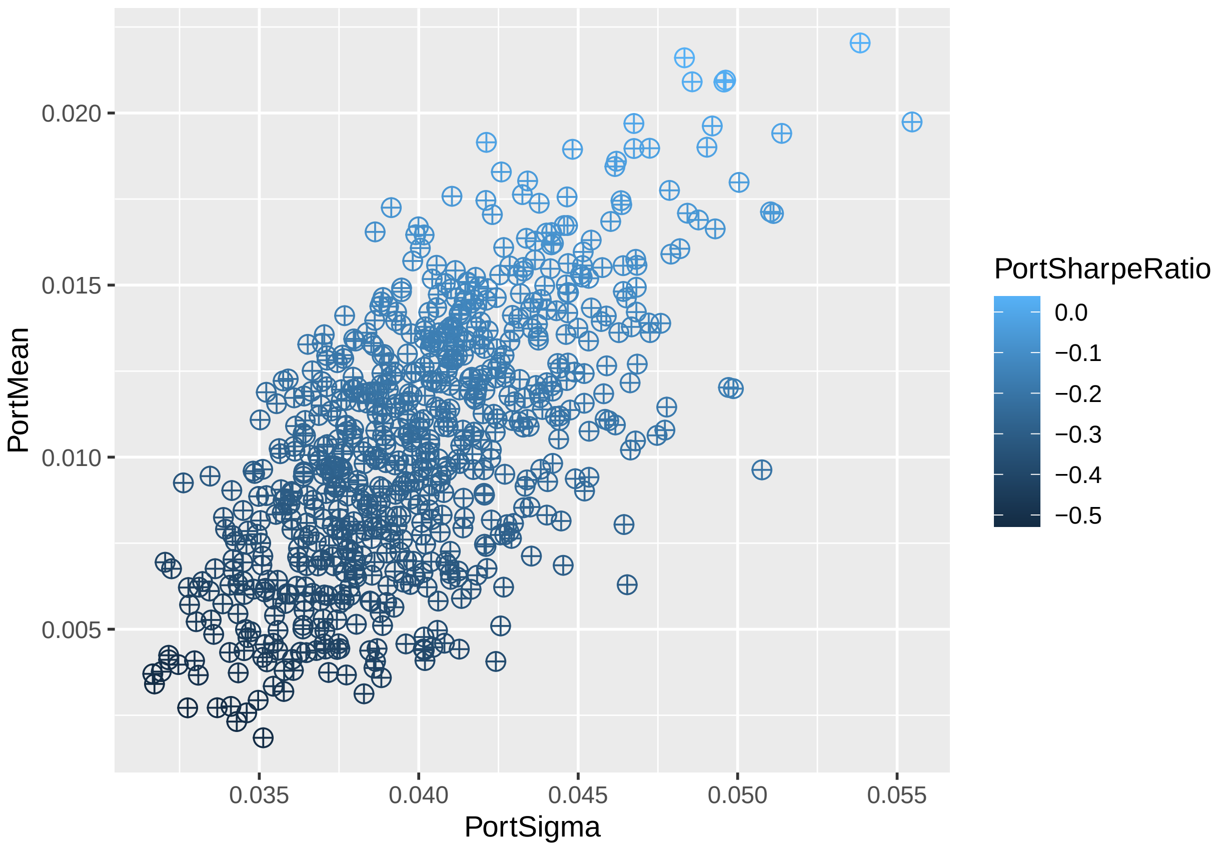
Let's show the result as a data frame that includes portfolio mean, sigma and Sharpe ratio and the plot the feasible set

```
Results <- output[["Results"]]
colnames(Results) <- c(ticks, "PortMean", "PortSigma", "PortSharpeRatio")
Results <- as.data.frame(Results)
head(Results)
```

```
##           GE        XOM        GBX       SBUX       PFE        HMC      NVDA
## 1 0.04591352 0.12500521 0.11204062 0.14103162 0.2409166 0.20463972 0.1304527
## 2 0.17032874 0.09825230 0.16522497 0.17753802 0.1241027 0.19185299 0.0727003
## 3 0.22918285 0.03894855 0.24130303 0.05836888 0.1482585 0.11881530 0.1651229
## 4 0.09390609 0.21051359 0.07800949 0.03641164 0.2498903 0.08883252 0.2424364
## 5 0.07590857 0.19030177 0.25140423 0.10023674 0.1593137 0.08202033 0.1408146
```

```
## 6 0.03183638 0.12317640 0.22950334 0.17884515 0.1092784 0.11006486 0.2172955
##      PortMean  PortSigma PortSharpeRatio
## 1 0.010364394 0.03713120     -0.25950161
## 2 0.006709984 0.03799179     -0.34981282
## 3 0.012159533 0.04662683     -0.16815356
## 4 0.015065803 0.04154418     -0.11876987
## 5 0.012226077 0.04463796     -0.17415499
## 6 0.017337207 0.04636429     -0.05743198
```

```
ggplot(data = Results , aes(x = PortSigma, y = PortMean, color = PortSharpeRatio)) +
  geom_point(pch = 10, size =3)
```



Let's print the mean returns, sigma and sharpe ration for the portfolio

```
weight<- output[["Portfolio_Weight"]]
covar <- output[["Covariance_Matrix"]]
meanret<- output[["Stock_means"]]
mretp <- t(weight) %*% meanret
sretp <- sqrt(t(weight) %*% covar %*% weight)
shrretp <- (mretp - rf_rate)/sretp

cat("The portfolio mean return, sigma and sharpe ratio: ", mretp, sretp,shrretp)
```

```
## The portfolio mean return, sigma and sharpe ratio:  0.003767694 0.03192729 -0.5084148
```

Optimization

We optimize the portfolios by maximizing sharpe ratio

```r
#Optimization
minmret = min(Results$PortMean)
maxmret = max(Results$PortMean)
seqmret = seq(round(minmret,3)-.001, maxmret+.001, .001)

optim <- Results %>% mutate(portnumber = index(Results)) %>%
  mutate(ints = cut(PortMean ,breaks = seqmret),
         lower = as.numeric( sub("\\((.+),.*", "\\1", ints) )) %>%
  group_by(ints) %>%
  summarise( lowerval = min(lower),
             sig_optim = min(PortSigma),
             ret_sigm_optim = PortMean[which.min(PortSigma)],
             ret= PortMean,
             retn_optim = PortMean[which.max(PortMean)],
             sr_optim = PortSharpeRatio[which.max(PortSharpeRatio)],
             numb = length(PortSigma),
             portID=portnumber[which.max(PortMean)])
```

```
## `summarise()` has grouped output by 'ints'. You can override using the
## `.groups` argument.
```

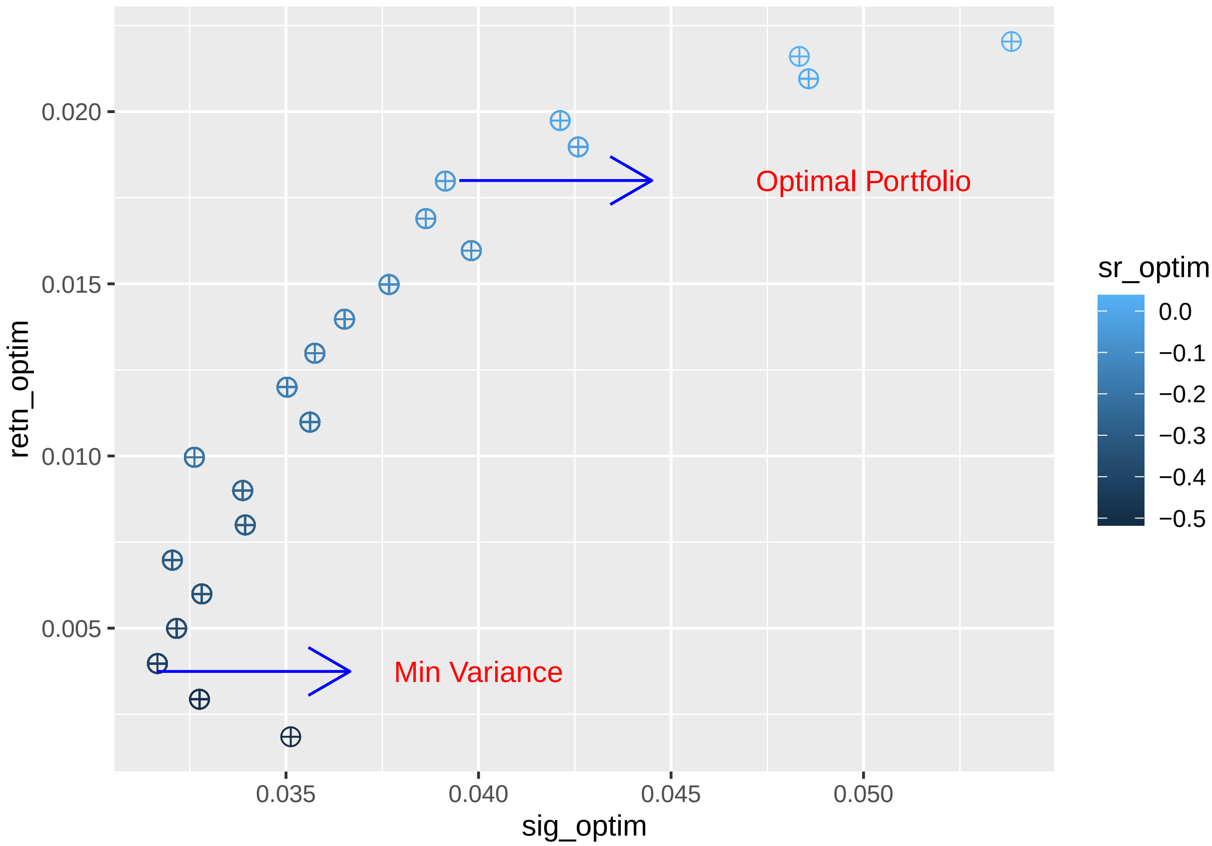
```r
head(optim)
```

```
## # A tibble: 6 x 9
## # Groups:   ints [2]
##   ints          lowerval sig_optim ret_si~1     ret retn_~2 sr_op~3  numb portID
##   <fct>            <dbl>     <dbl>    <dbl>   <dbl>   <dbl>   <dbl> <int>  <int>
## 1 (0.001,0.002]    0.001    0.0351  0.00185 0.00185 0.00185  -0.517     1    350
## 2 (0.002,0.003]    0.002    0.0328  0.00272 0.00276 0.00294  -0.488     6    216
## 3 (0.002,0.003]    0.002    0.0328  0.00272 0.00272 0.00294  -0.488     6    216
## 4 (0.002,0.003]    0.002    0.0328  0.00272 0.00272 0.00294  -0.488     6    216
## 5 (0.002,0.003]    0.002    0.0328  0.00272 0.00294 0.00294  -0.488     6    216
## 6 (0.002,0.003]    0.002    0.0328  0.00272 0.00232 0.00294  -0.488     6    216
## # ... with abbreviated variable names 1: ret_sigm_optim, 2: retn_optim,
## #   3: sr_optim
```

Plot the optimized potfolios

```r
xcoord_minvar <- min(optim$sig_optim)
ycoord_minvar<- optim$ret[which.min(optim$sig_optim)]


ggplot(data = optim , aes(x = sig_optim, y = retn_optim, color = sr_optim)) +
  geom_point(pch = 10, size = 3) +
  annotate("segment", x = xcoord_minvar, y = ycoord_minvar,
         xend = xcoord_minvar + .005, yend =ycoord_minvar ,
         arrow=arrow(), color = "blue") +
  annotate("text", x = .040, y = ycoord_minvar, label = 'Min Variance', color = 'red')+
  annotate("segment", x = 0.0395, y = 0.018,
           xend = 0.0395+ .005, yend =0.018 ,
           arrow=arrow(), color = "blue") +
  annotate("text", x = .050, y = 0.018, label = 'Optimal Portfolio', color = 'red')
```



Conclusion:

Monte-Carlo Simulation is one of the methods typically used to optimize portfolios. There is however no one correct method or result as the paramters may differ and it involves some speculation. That said, let's looks at some advantages and disadvantages of this method:

Advantages:

The biggest advantage of this method is its ability to factor in a range of values for various inputs converting chances into choices. It can also be easily plotted for visual aid.

Disadvantages:

Since it depends on inputs it needs the assumptions to be fair Another disadvantage is that ot tends to underestimate the behaviorial aspect of finance and the irrationality demonstrated by participants.

For all its pros and cons, Monte-Carlo simulation remains one of the most widely used methods in finance for optimization and is a useful tool for advisors.