# Project1 Part2 Applied Statistics

Abhina Premachandran

2023-11-26

This project uses Monte Carlo Simulations to simulate daily returns of an investment portfolio. An optimal portfolio of two different stocks from a list of five stocks('DIS','HPQ','USO','DVN','LMT') is found by calculating the minimum risk weights and returns of the possible pairs. `getsymbols` function of the `quantmod` library is used to get the stock values. The daily return values of each stocks are found using diff()/lag() of the closing prices. Further, the mean and standard deviations of these daily returns is used for finding the simulated returns using Monte Carlo Simulation.

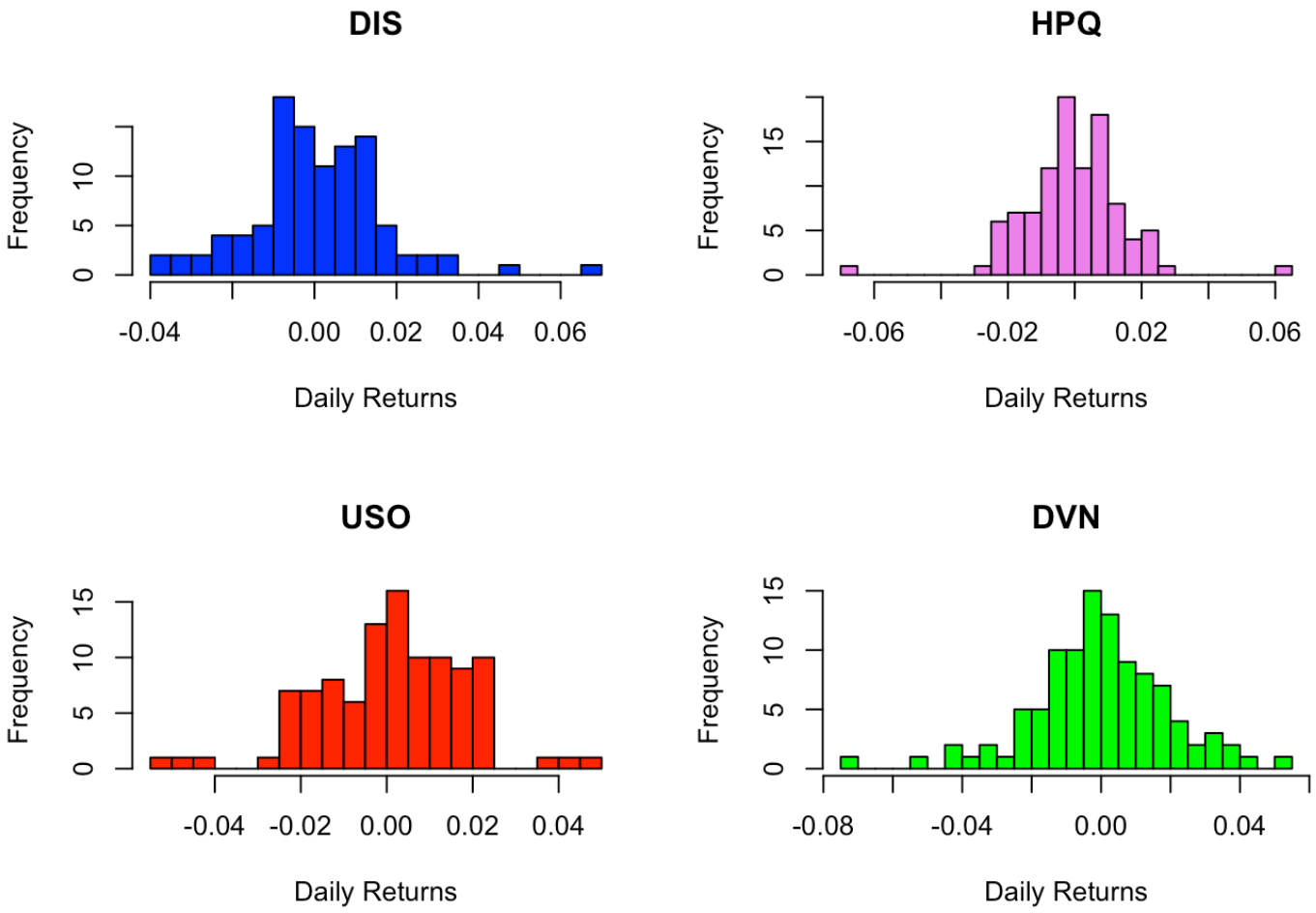
# Getting the stock values, finding the return, daily mean and daily std for 'DIS','HPQ','USO', 'DVN','LMT' stocks

Getting the stock data using `getSymbols`.

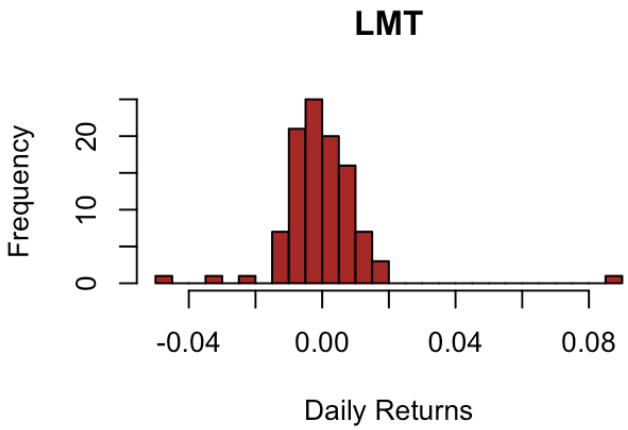
```
#intializing the lists
return_list <- list()
daily_mean <- list()
daily_std <- list()
starting_price_ <- list()
total_returns <- list()
sim_returns <- list()


ticker <- c('DIS','HPQ','USO','DVN','LMT')
for(i in ticker){
  #getting the stock values for 150 days until today
  x <- getSymbols(i,from = Sys.Date() - 150, to = Sys.Date(),auto.assign = FALSE)

  return_list[[i]] <- diff(Cl(x))/lag(Cl(x))
  N_days <- length(return_list[[i]])

  # Finding the mean and std deviation of stocks
  m <- mean(return_list[[i]][(N_days - 31):N_days])
  daily_mean[[i]] <- m
  s <- sd(return_list[[i]][(N_days - 31):N_days])
  daily_std[[i]] <- s


  #Monte Carlo Simulation for stocks
  set.seed(101)
  start_price <- Cl(x)[[N_days-30]]
  starting_price_[[i]] <- start_price

  sim_returns[[i]] <- 1+rnorm(150,mean=m,sd=s)
  no_sims <- 1001
  days_forward <- 30
  sim_return_list <- matrix(0, nrow=no_sims,ncol=days_forward)
  sim_price_list <- matrix(0, nrow=no_sims,ncol=days_forward+1)
  for (j in 1:no_sims) {
     sim_return_list[j,] <- rnorm(days_forward, mean = m, sd = s)
     sim_price_list[j,] <- cumprod(c(start_price, 1+sim_return_list[j,]))
  }

  for(j in 1:no_sims){
  total_returns[[i]][j]<-(sim_price_list[j,days_forward+1]-sim_price_list[j,1])/sim
_price_list[j,1]
  }
}
```

# Visualizing the returns of stocks

```
par(mfrow = c(2, 2))
hist(return_list[['DIS']],main = 'DIS', xlab = 'Daily Returns', col='blue',breaks =
20, freq=TRUE)
hist(return_list[['HPQ']],main = 'HPQ', xlab = 'Daily Returns', col='violet',breaks
= 20, freq=TRUE)
hist(return_list[['USO']],main = 'USO', xlab = 'Daily Returns', col='red',breaks =
20, freq=TRUE)
hist(return_list[['DVN']],main = 'DVN', xlab = 'Daily Returns', col='green',breaks
= 20, freq=TRUE)
```




```
hist(return_list[['LMT']],main = 'LMT', xlab = 'Daily Returns', col='brown',breaks
= 20, freq=TRUE)
```



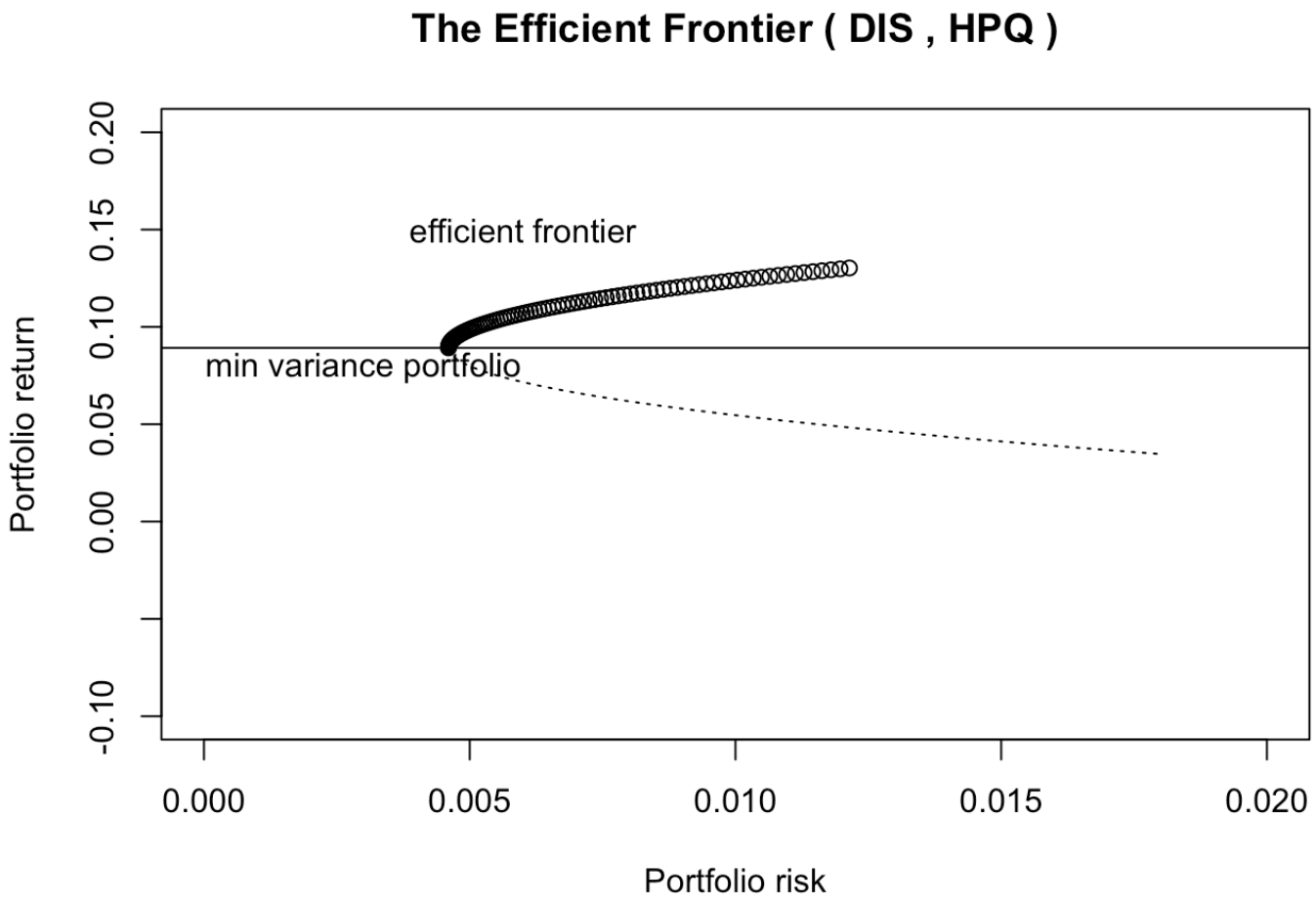

From the histogram of each stock, it is clear that the daily returns of the stocks approximately follows a normal distribution. Therefore rnorm() can be used to simulate future returns.

# Building an optimal portfolio

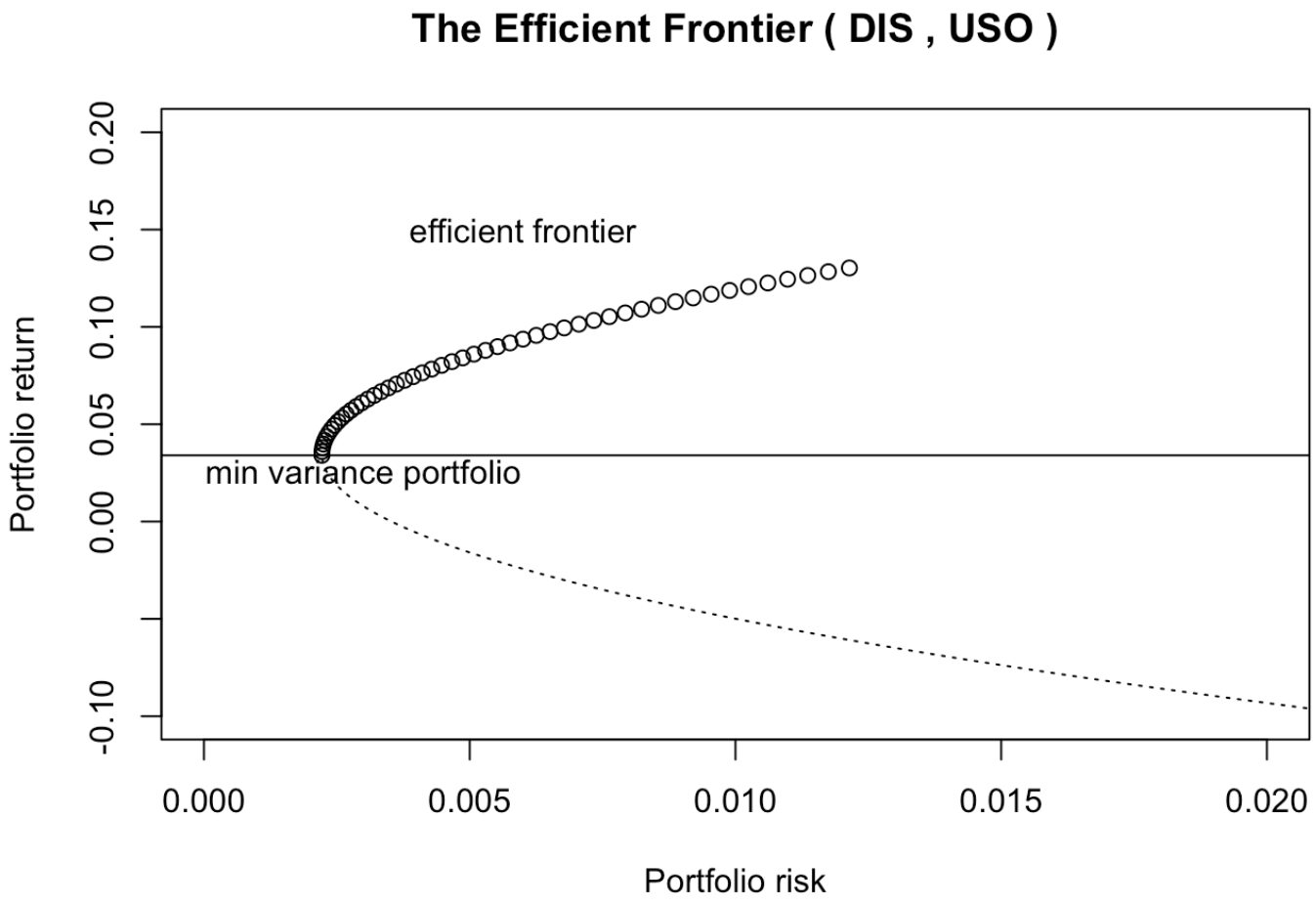
To build a portfolio, the return and risks for different values of weights are calculated using the mean and standard deviations from the simulated returns of stocks.

```
ticker <- c('DIS','HPQ','USO','DVN','LMT')

for(x in 1:(length(ticker) - 1)){
  s1 <- getSymbols(ticker[x],from = Sys.Date() - 150, to = Sys.Date(),auto.assign =
FALSE)
  mu1 <- mean(total_returns[[x]])
  sigma1 <- sd(total_returns[[x]])
  for(y in (x+1):length(ticker)){
    s2 <- getSymbols(ticker[y],from = Sys.Date() - 150, to = Sys.Date(),auto.assign
= FALSE)
    mu2 <- mean(total_returns[[y]])
    sigma2 <- sd(total_returns[[y]])
    ClosePrices <- merge(Cl(s1),Cl(s2))
    # Calculating the correlation coefficient(rho) of each pairs of stocks
```

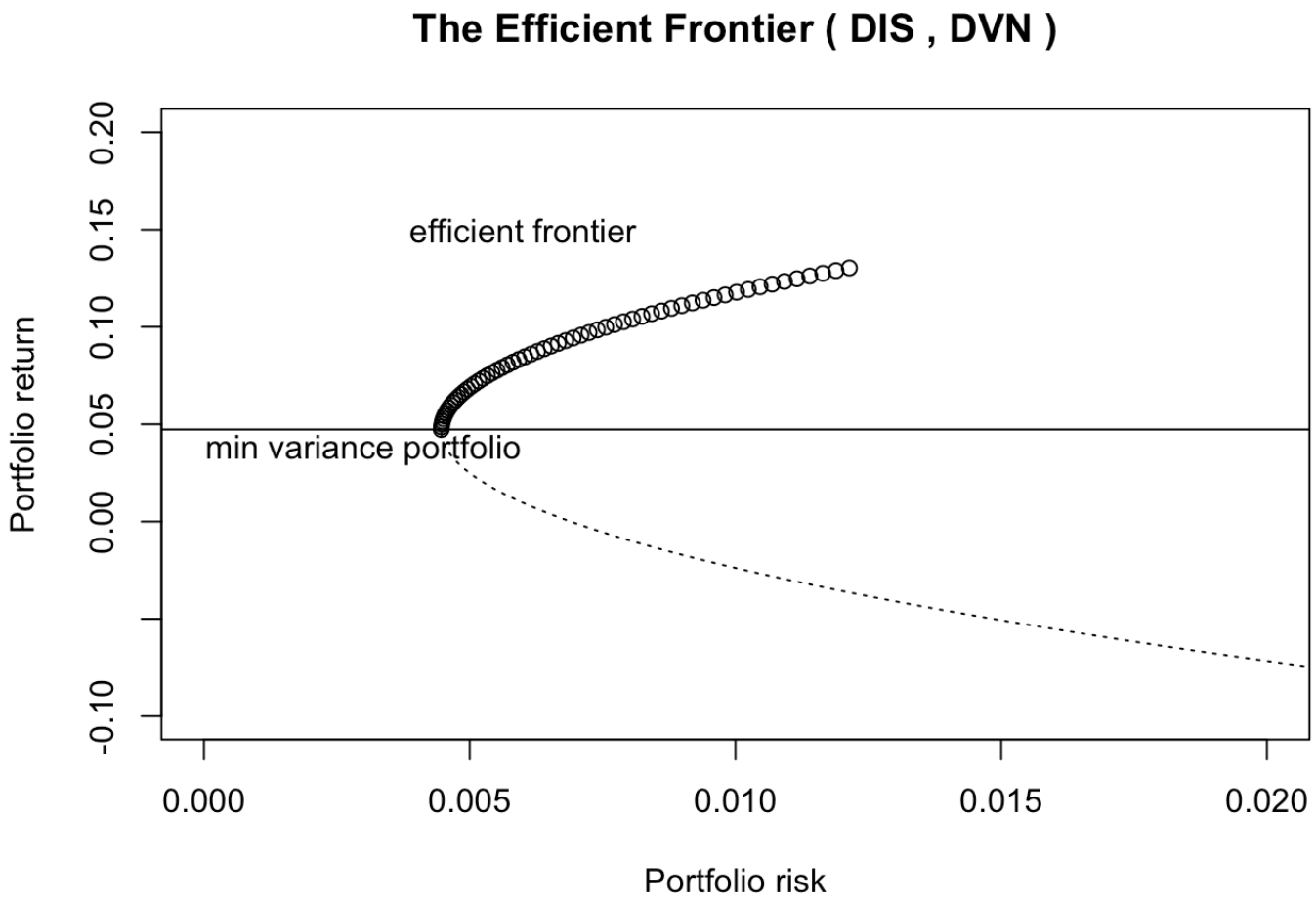
```
    rho_12 <- cor(ClosePrices, use = "pairwise.complete.obs")[1,2]

    # finding the total return and total risk
    total_return <- function(w){
      return(w*mu1 + (1-w)*mu2)
    }
    total_risk <- function(w){
      return(w^2*sigma1^2 + (1-w)^2*sigma2^2 + 2*w*(1-w)*rho_12*sigma1*sigma2)
    }

    # taking all values from -1 to 1 with a step value of 0.01 for weights
    weights <- seq(-1, 1, by = 0.01)
    returns <- NULL
    risks <- NULL
    #finding the return and risk values for each weights
    for(w in weights){
      returns <- c(returns, total_return(w))
      risks <- c(risks, total_risk(w))
    }
    #finding the minimum risk weight and the corresponding return
    min_risk_w <- weights[which(risks == min(risks))]
    min_risk_return <- mean(total_return(min_risk_w))
    efficient_idx <- which(returns >= min_risk_return)
    inefficient_idx <- which(returns < min_risk_return)

    # Plotting the efficient frontier for the stock pairs
    plot(risks[efficient_idx], returns[efficient_idx],
    xlab = "Portfolio risk",
    ylab = "Portfolio return",
    main = paste("The Efficient Frontier (",ticker[x],",",ticker[y],")"),
    ylim = c(-0.1,0.2), xlim = c(0.00, 0.02))

    lines(risks[inefficient_idx], returns[inefficient_idx], lty = "dotted")

    abline(h = min_risk_return)
    text(x = 0.003, y = min_risk_return-0.01, "min variance portfolio")
    text(x = 0.006, y = 0.15, "efficient frontier")
    print(paste("(",ticker[x],",",ticker[y],"), Rho_12:",round(rho_12,digits=4),",m
in_risk_w:",round(min_risk_w,digits=4),",min_risk_return:",round(min_risk_return,di
gits=4)))

  }
}
```



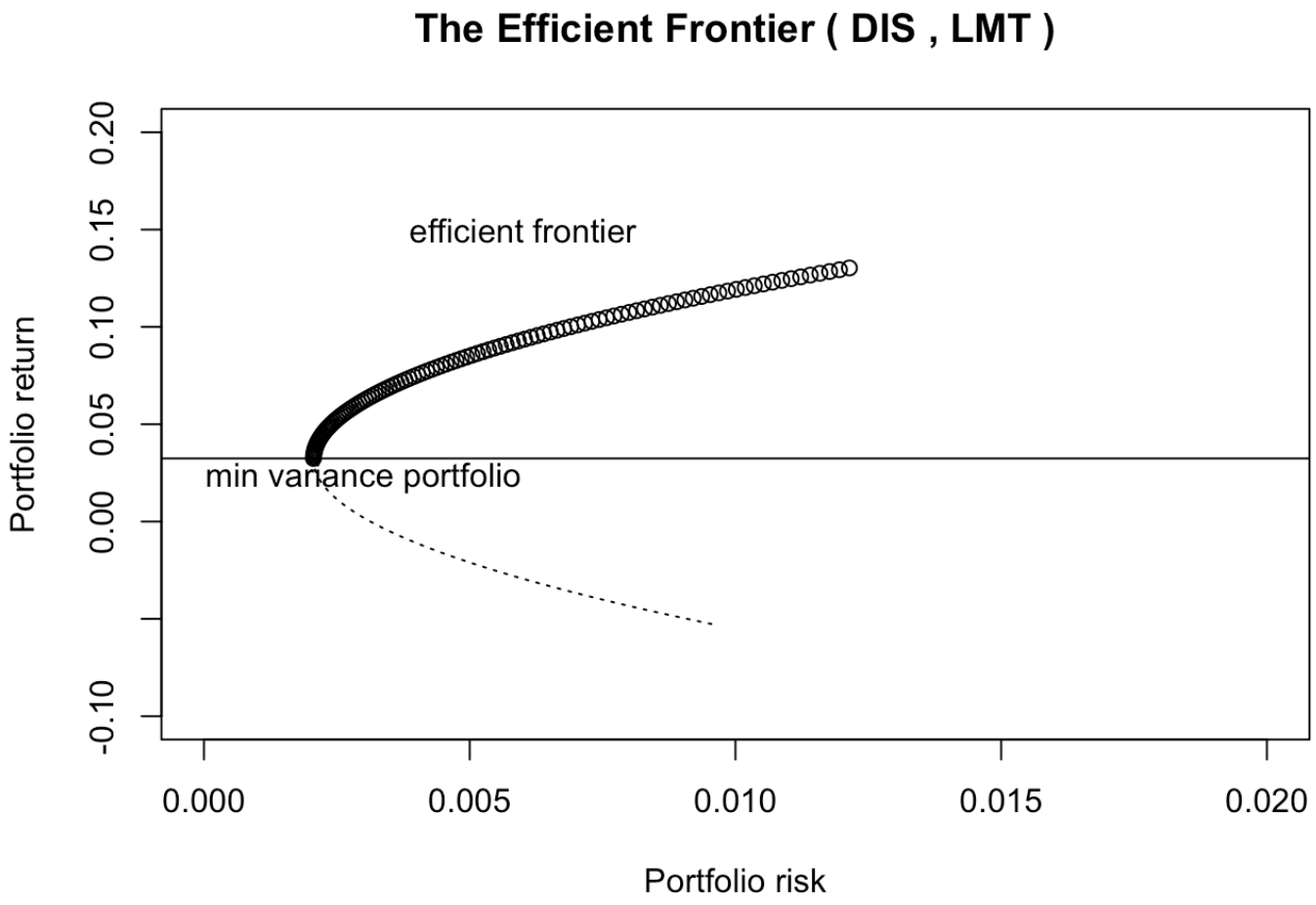

```
## [1] "( DIS , HPQ ), Rho_12: 0.4397 ,min_risk_w: 0.14 ,min_risk_return: 0.0892"
```



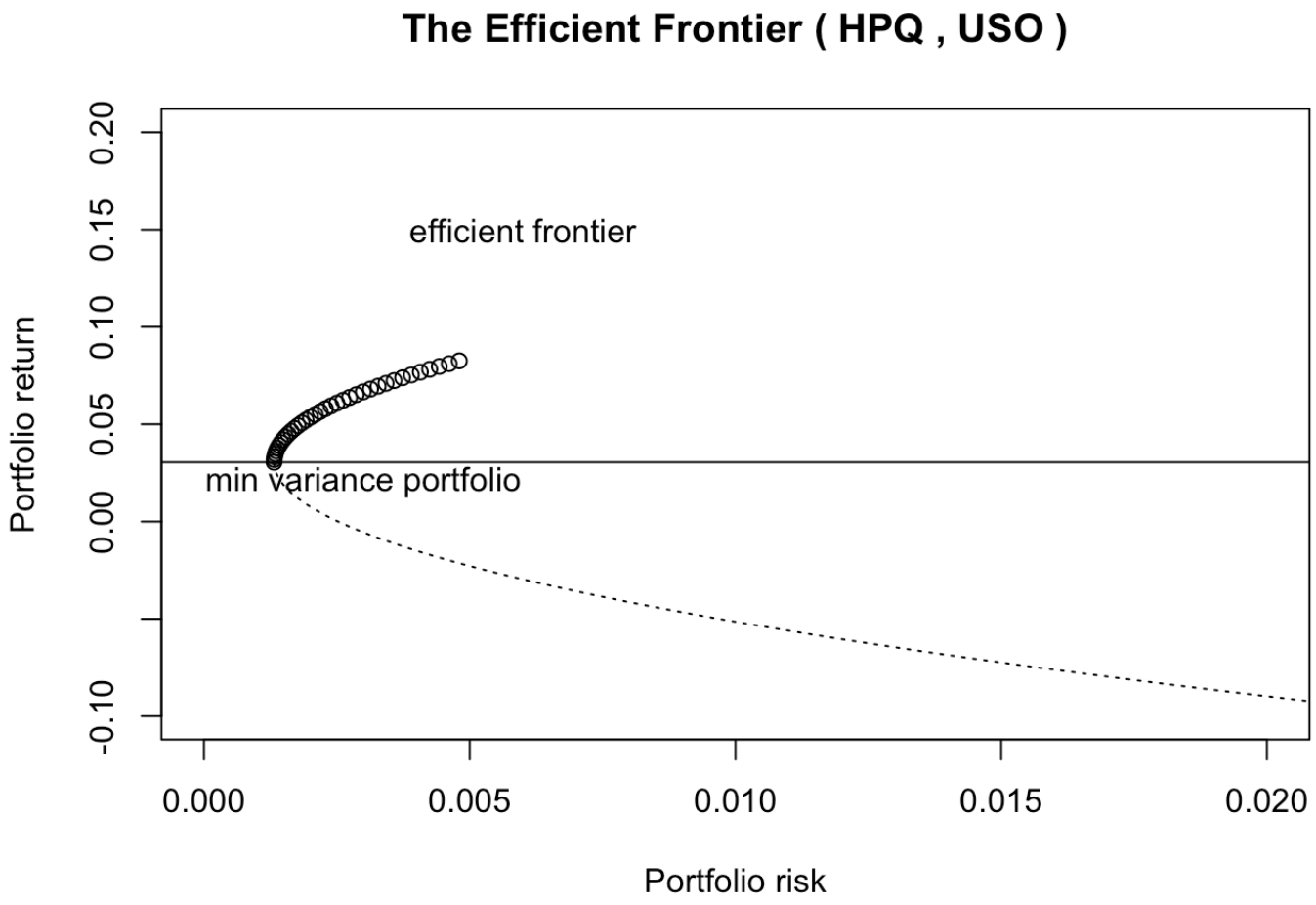

```
## [1] "( DIS , USO ), Rho_12: -0.6387 ,min_risk_w: 0.5 ,min_risk_return: 0.034"
```



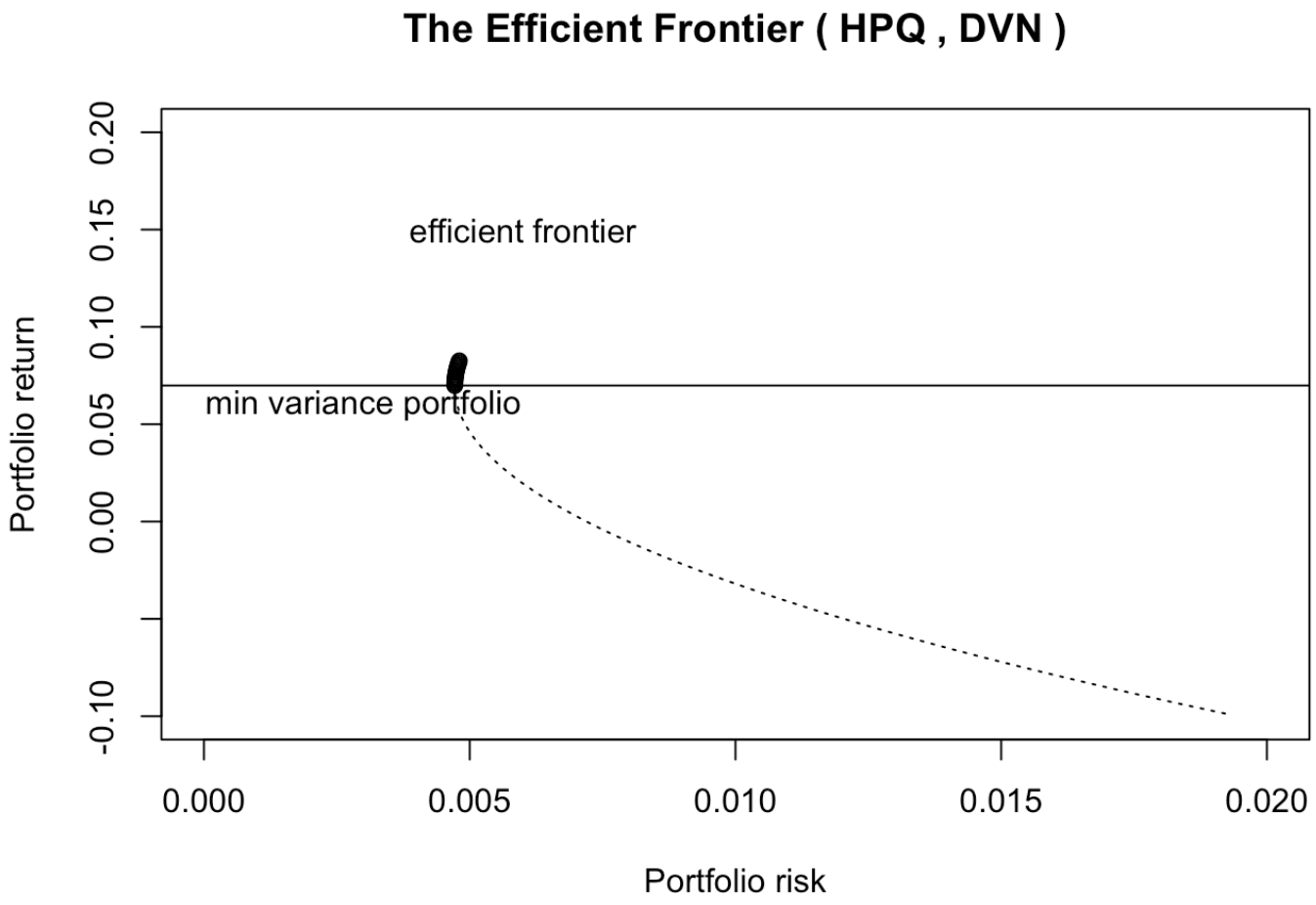

```
## [1] "( DIS , DVN ), Rho_12: -0.0616 ,min_risk_w: 0.4 ,min_risk_return: 0.0473"
```



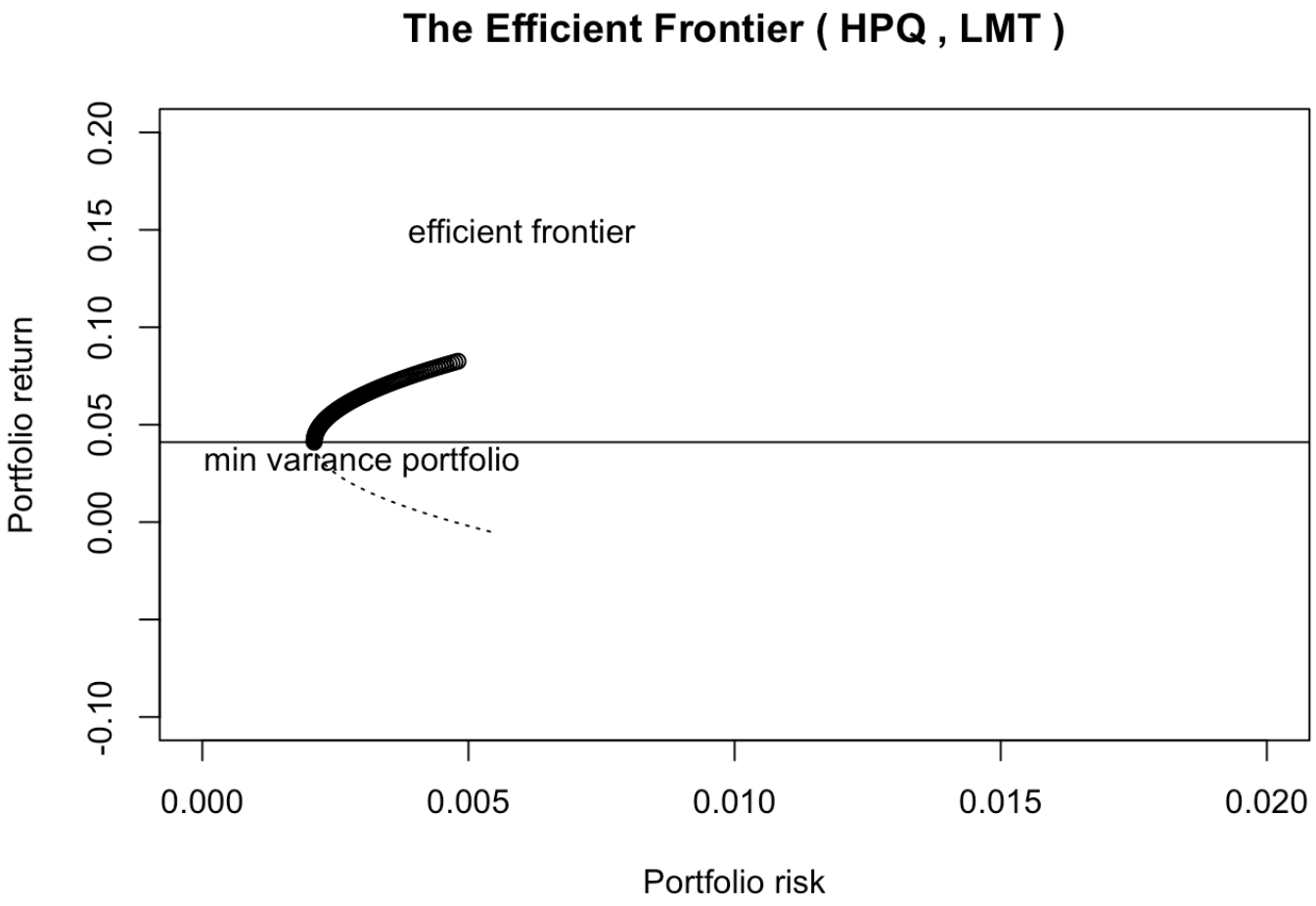

```
## [1] "( DIS , LMT ), Rho_12: 0.5449 ,min_risk_w: -0.07 ,min_risk_return: 0.0324"
```



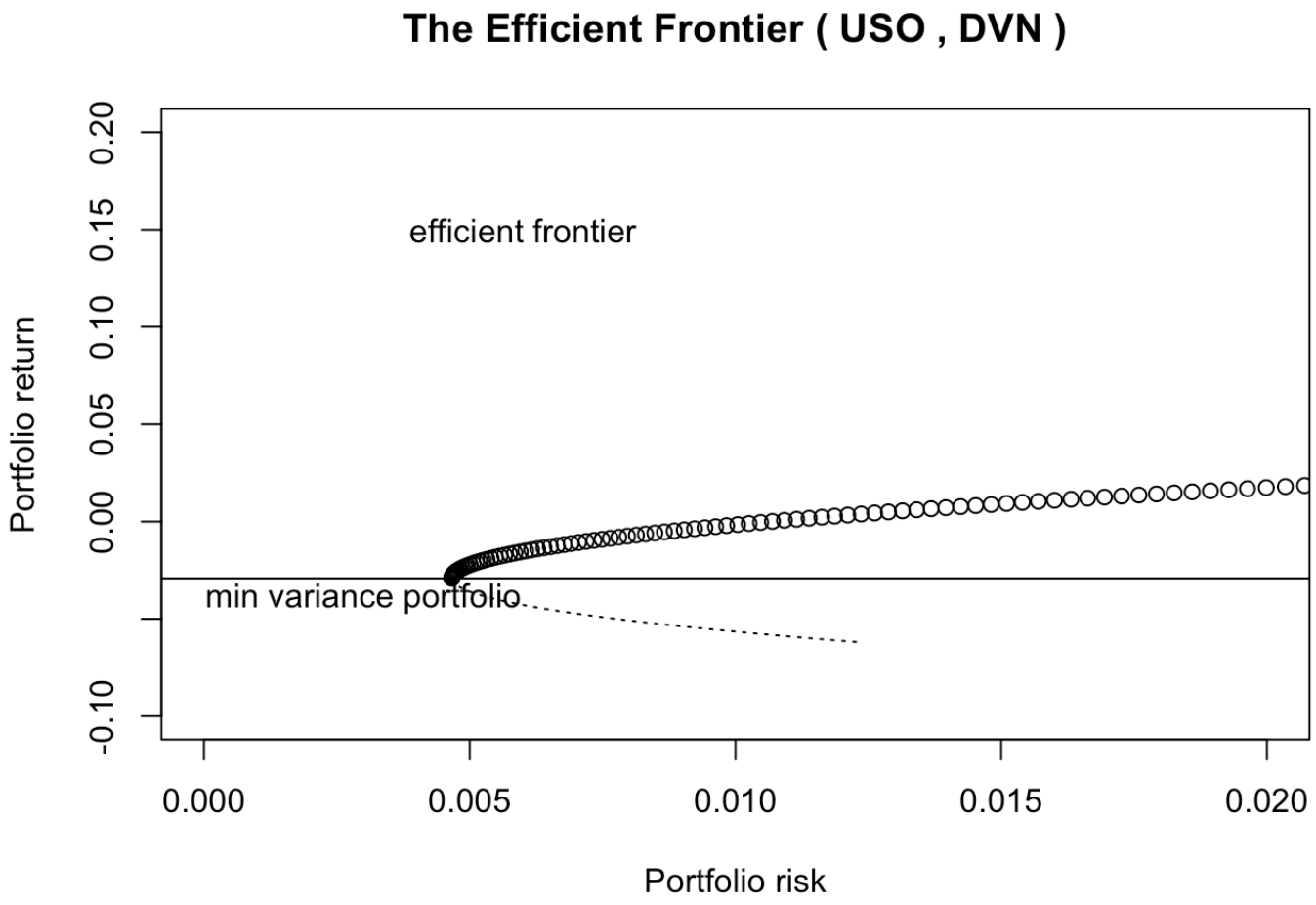

```
## [1] "( HPQ , USO ), Rho_12: -0.6345 ,min_risk_w: 0.64 ,min_risk_return: 0.0304"
```



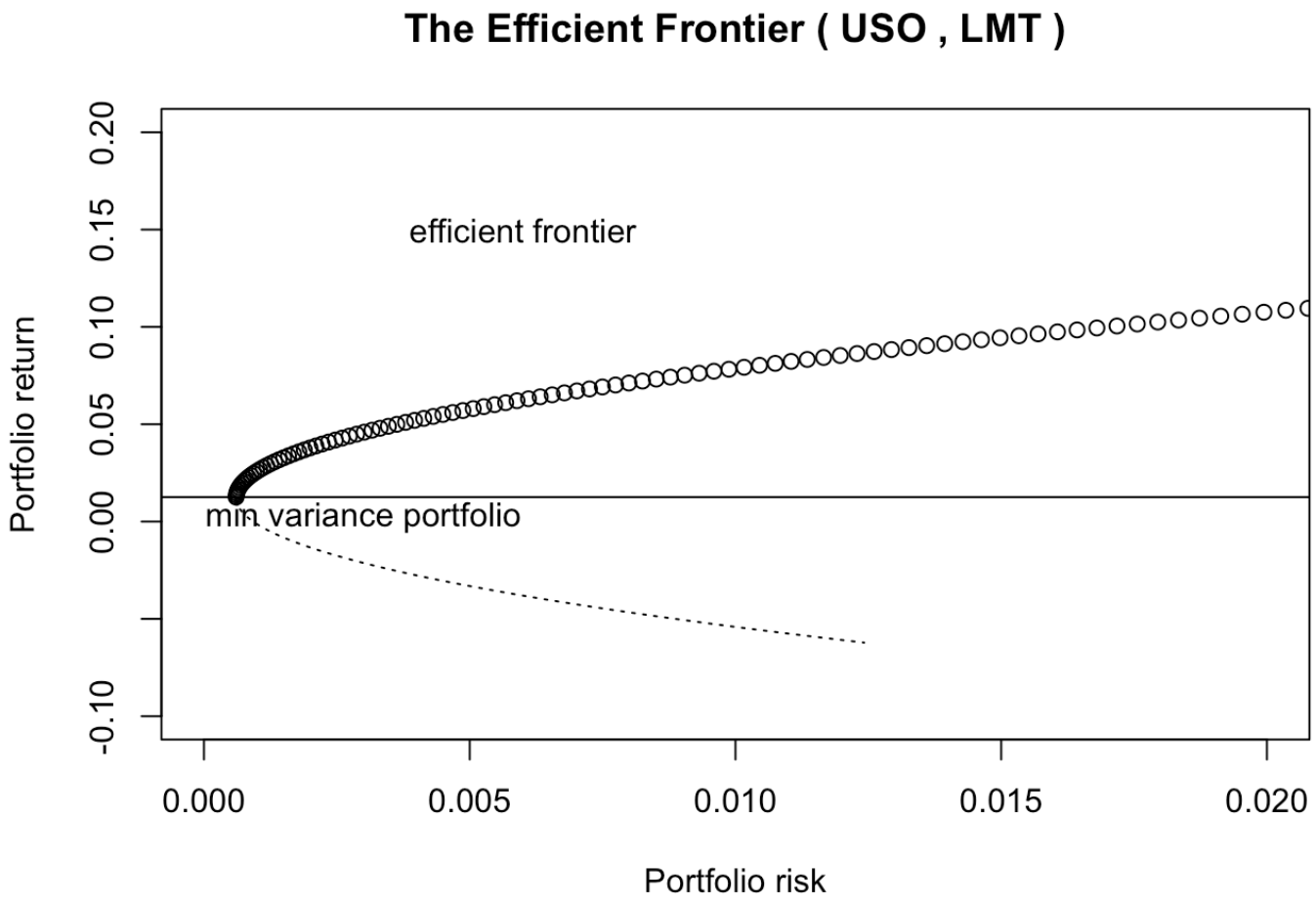

```
## [1] "( HPQ , DVN ), Rho_12: 0.6863 ,min_risk_w: 0.86 ,min_risk_return: 0.0699"
```



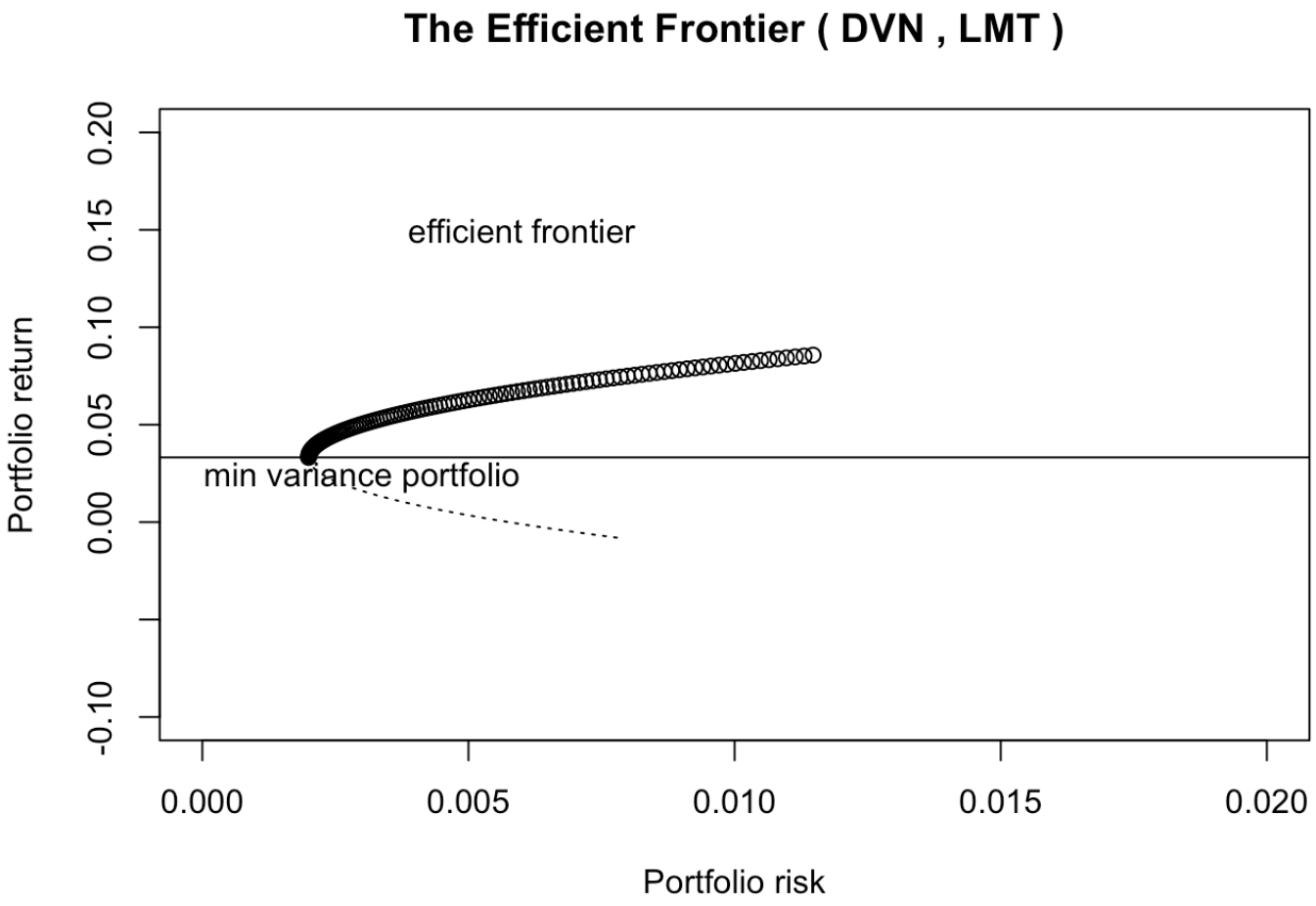

```
## [1] "( HPQ , LMT ), Rho_12: 0.6149 ,min_risk_w: 0.05 ,min_risk_return: 0.041"
```




```
## [1] "( USO , DVN ), Rho_12: -0.028 ,min_risk_w: 0.39 ,min_risk_return: -0.0292"
```




```
## [1] "( USO , LMT ), Rho_12: -0.7059 ,min_risk_w: 0.26 ,min_risk_return: 0.0126"
```




```
## [1] "( DVN , LMT ), Rho_12: 0.2934 ,min_risk_w: 0.12 ,min_risk_return: 0.0332"
```

From the efficient frontier of the return values of all ten pairs of stocks, (DIS,HPQ) stocks have the highest return value of $0.0892 with a minimum risk weight of 14% for DIS and 86% for HPQ stock. They have a positive correlation coefficient value of 0.4397, which means they either increases or decreases together in stock value. A portfolio including the USO and DVN stocks gives a negative minimum risk return value of -0.0292 which is not beneficial. The best pair of stocks for getting the highest minimum risk return value can be found using this method. But need to check the accuracy of the simulations used. This is done in the next section.

# Evaluation

```
# comparing the actual returns and simulated returns to evaluate the accuracy of si
mulation
ticker <- c('DIS','HPQ','USO','DVN','LMT')
per_error <- list()
for(i in ticker){
  x <- mean(return_list[[i]][2:104])#the first value is excluded as it is NA
  y <- mean(total_returns[[i]][2:104])#the first value is excluded as it is NA
  per_error[[i]] <- abs(((x-y)/x))
  print(paste('percentage error of ',i,' is ',per_error[[i]]))
}
```

```
## [1] "percentage error of  DIS  is  156.742260056676"
## [1] "percentage error of  HPQ  is  198.535231450886"
## [1] "percentage error of  USO  is  42.3981155215655"
## [1] "percentage error of  DVN  is  2.00084383623624"
## [1] "percentage error of  LMT  is  1238.1992728137"
```

For evaluating the accuracy of results, the percentage error between the mean values of actual and simulated returns is found. Since the first value of actual and simulated returns is NA, percentage error is calculated for values from 2 to the length of actual returns(104). From the values found, ‘LMT’ have a very high percentage error value of 1238 % and ‘DVN’ have a low percentage error value of 2 %. This indicates that the simulation method used does not work well for all stocks. There might be some other factors that affect the trend of stocks which is causing the deviation of actual price values from the predicted ones.

# Conclusion

From the analysis above, it is clear that the Monte Carlo simulation used for simulating the returns is not working as desired for some stocks in the list. Therefore, the optimal portfolio found above might have some errors as these simulated returns are used for finding the minimum risk weight returns for each pair. Overall, this is a good starting point. The technique used for building the optimal portfolio is reliable, provided the simulated returns are similar to the actual returns. Therefore, it can be concluded that the Monte Carlo simulation and using it to build an optimal portfolio works well for stocks that have a trend comparable to a normal distribution.