```
rm(list=ls(all=TRUE))
setwd("/home/dev/work/Insofe/Mini-Project-2/ShinyStockPortfolio")

library(quadprog)

# Select no of stocks
num.stocks <- 5
# Load stock data for randomly selected num.stocks
source("LoadStockData.R")

#Selected Stocks
sel.stocks
```

```
## [1] "DLF.csv"           "Ranbaxy.csv"       "Maruti Suzuki.csv"
## [4] "SBI.csv"           "Asian Paints.csv"
```

```
#Create asset returns which will be used by portfolio
asset.returns <- data.frame()
for (i in 1:num.stocks) {
  day.returns <- stocks[[i]]$Day.Return
  if (i == 1) {
    asset.returns <- data.frame(day.returns)
  } else {
    asset.returns <- cbind(asset.returns,day.returns)
  }
  i <- i+1
}

# Function to evaluate the minimum risk given a required return
portfolio <- function(assetReturns, targetReturn)
{
  # Arguments:
  # assetReturns - multivariate data set of asset returns
  # target Return - the portfolios target return
  # 1 Create Portfolio Settings:
  nAssets = ncol(assetReturns)
  Dmat = cov(assetReturns)
```

```
  dvec = rep(0, times=nAssets)
  Amat = t(rbind(
    Return=colMeans(assetReturns),
    Budget=rep(1, nAssets),
    LongOnly=diag(nAssets)))
  bvec = c(
    Return=targetReturn,
    budget=1,
    LongOnly=rep(0, times=nAssets))
  meq = 2
  # 2 Optimize Weights:
  portfolio = solve.QP(Dmat, dvec, Amat, bvec, meq)
  weights = round(portfolio$solution, digits = 4)
  names(weights) = colnames(assetReturns)
  # Return Value:
  list(
    weights = 100*weights,
    risk = portfolio$value,
    return = targetReturn)
}

#Invoke the portfolio function for different returns starting
#at 1% and incrementing by 0.1%
ret.reqd <- 0.01
risk.returns <- data.frame()
# Loop until returns are 100% (very unlikely to reach) OR the QP fails
# which is more likely
while (ret.reqd < 1) {
  tryCatch(
    #Handle Errors - Error message expected
    pfolio <- portfolio(asset.returns,ret.reqd), error=function(e) break
  )
  risk.returns <- rbind(risk.returns, c(pfolio$risk, pfolio$return))
  #1% increments to required return
  ret.reqd <- ret.reqd+(0.001)
}
```

```
## Error: no loop for break/next, jumping to top level
```

```r
colnames(risk.returns) <- c("Risk", "Returns")
head(risk.returns,10)
```

```
##     Risk Returns
## 1  1.367   0.010
## 2  1.357   0.011
## 3  1.347   0.012
## 4  1.337   0.013
## 5  1.327   0.014
## 6  1.318   0.015
## 7  1.310   0.016
## 8  1.301   0.017
## 9  1.293   0.018
## 10 1.285   0.019
```

```r
risk.min <- min(risk.returns$Risk)
risk.min.ret <- risk.returns[which(risk.returns$Risk==risk.min),]$Returns
cat("Minimum Risk: ", risk.min, ", at Return: ", risk.min.ret)
```

```
## Minimum Risk:  1.205 , at Return:  0.04
```

```r
#Plot the returns to risk efficiency frontier
plot(risk.returns, type='l', col="blue", main="Risk-Return Efficiency Frontier")
abline(v=risk.min,h=risk.min.ret,col="red",lty=2)
```