5261project

Jiayi Yuan

2022-04-20

Descriptive Statistics

```r
library(readxl)
library(moments)
library(reshape)
library(corrgram)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following objects are masked from 'package:reshape':
##
##     expand, smiths
```

```r
library(tseries)
```

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

```r
library(fGarch)
```

```
## Loading required package: timeDate

##
## Attaching package: 'timeDate'

## The following objects are masked from 'package:moments':
##
##     kurtosis, skewness

## Loading required package: timeSeries

## Loading required package: fBasics
```

```r
asset <- read_xlsx("12Assetdata.xlsx", sheet = "Price")
return <- read_xlsx("12Assetdata.xlsx", sheet = "Return")
asset[,-1] <- round(asset[,-1], 4)
asset$Date <- as.Date(asset$Date)
return$Date <- as.Date(return$Date)
```

Means SDs Skewness Kurtosis Betas

```r
means <- sapply(asset[,2:14], mean)
means
```

```
##       MSFT       TSLA       AAPL       TWTR       AMZN         FB       NFLX
##  117.50850  203.20226   58.86989   34.92101 1550.22788  171.00727  264.86528
##        AAL        DAL        BAC       NVDA        WBD     S&P500
##   33.15034   42.93120   23.75602   64.03213   29.02633 2770.63828
```

```
means_r <- sapply(return[,2:14], mean)
means_r
```

```
##        MSFT        TSLA        AAPL        TWTR        AMZN          FB
## 0.024946532 0.048015901 0.027737763 0.005061345 0.025953512 0.016357391
##        NFLX         AAL         DAL         BAC        NVDA         WBD
## 0.025691372 0.001117527 0.008395770 0.013984216 0.051459413 0.001060357
##      S&P500
## 0.010358383
```

```
### SDs
sds <- sapply(asset[,2:14], sd)
sds
```

```
##        MSFT        TSLA        AAPL        TWTR        AMZN          FB
##   86.816870  298.244528   45.183336   14.228913 1062.254285   80.592563
##        NFLX         AAL         DAL         BAC        NVDA         WBD
##  176.639466   11.579518    8.402903    9.368711   74.026077    6.327468
##      S&P500
##  799.620131
```

```
sds_r <- sapply(return[,2:14], sd)
sds_r
```

```
##       MSFT       TSLA       AAPL       TWTR       AMZN         FB       NFLX
## 0.05917747 0.17296293 0.07743821 0.14535005 0.08163112 0.08122604 0.11741366
##        AAL        DAL        BAC       NVDA        WBD     S&P500
## 0.11493264 0.09510284 0.08191269 0.11778168 0.11332111 0.04002103
```

```
### Skewness
skews <- sapply(asset[,2:14], skewness)
skews
```

```
##        MSFT        TSLA        AAPL        TWTR        AMZN          FB
##  1.02751185  1.79579212  1.25136725  0.63979815  0.54077583  0.70211095
##        NFLX         AAL         DAL         BAC        NVDA         WBD
##  0.44730875 -0.35031528 -0.05642907  0.68530412  1.67311570  1.07671210
##      S&P500
##  0.95119157
```

```
skews_r <- sapply(return[,2:14], skewness)
skews_r
```

```
##        MSFT        TSLA        AAPL        TWTR        AMZN          FB
##  0.22793365  1.27334807 -0.22966311  0.41006779  0.39545316 -0.32966287
##        NFLX         AAL         DAL         BAC        NVDA         WBD
##  0.46509511 -0.09465125 -0.18437061 -0.15661734  0.06986737  0.84068916
##      S&P500
## -0.38746520
```

```
### Kurtosis
kurtosis <- sapply(asset[,2:14], kurtosis)
kurtosis
```

```
##         MSFT         TSLA         AAPL         TWTR         AMZN           FB
## -0.138676463  1.873876650  0.225458829 -0.004106557 -1.090615618 -0.295180805
##         NFLX          AAL          DAL          BAC         NVDA          WBD
## -1.036773152 -0.941063372 -0.647993063 -0.320107115  2.157497699  1.362829418
##       S&P500
## -0.110939419
```

```
kurtosis_r <- sapply(return[,2:14], kurtosis)
kurtosis_r
```

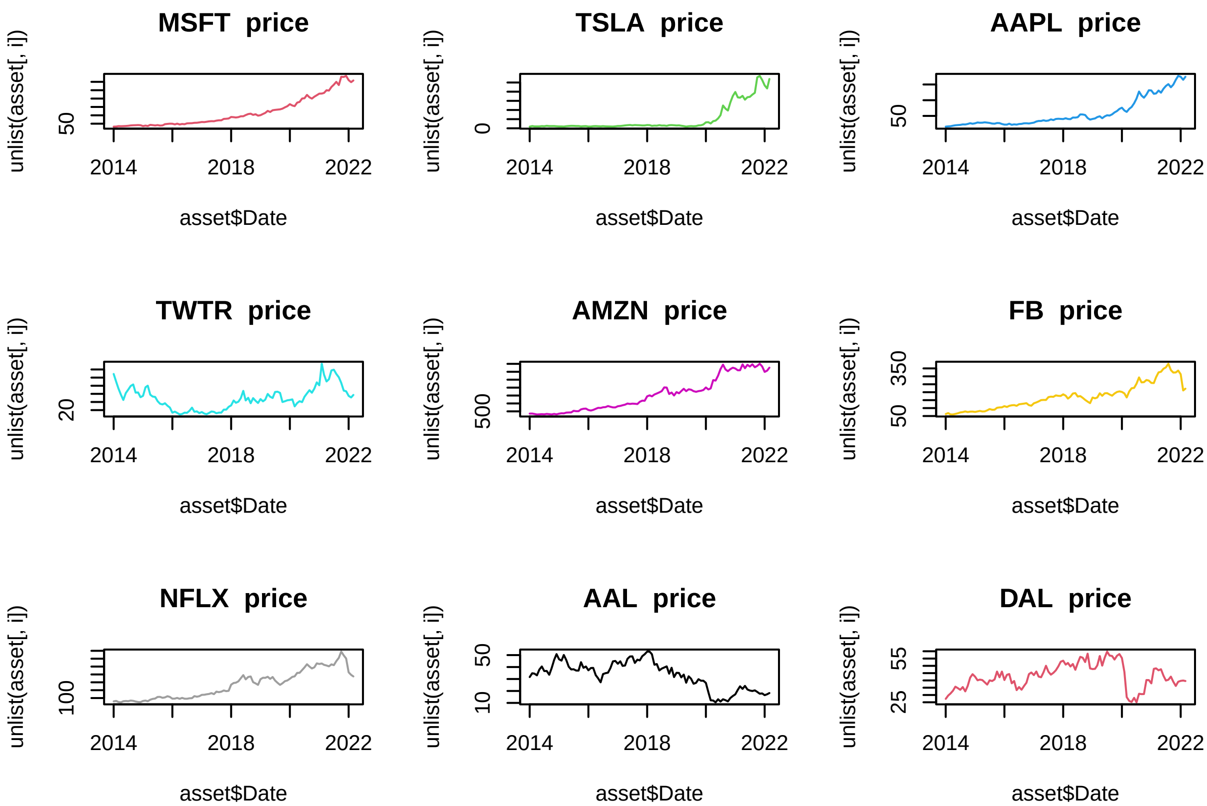
```
##       MSFT       TSLA       AAPL       TWTR       AMZN         FB       NFLX
##  0.6159448  2.2335358 -0.2720155  0.3760462  0.6509264  2.7725659  1.0312261
##        AAL        DAL        BAC       NVDA        WBD     S&P500
##  0.2174945  2.6293047  1.0251537  0.4181741  1.1073195  1.3428217
```

```
### Betas
betas <- list()
for (i in 2:13){
  betas[i-1] <- lm(unlist(return[,i])-return$`Treasury Bill 3 month (rf)`~
                return$`S&P500`- return$`Treasury Bill 3 month (rf)`)$coefficients[2]
}
names <- colnames(asset)[2:13]
rbind(names, unlist(betas))
```

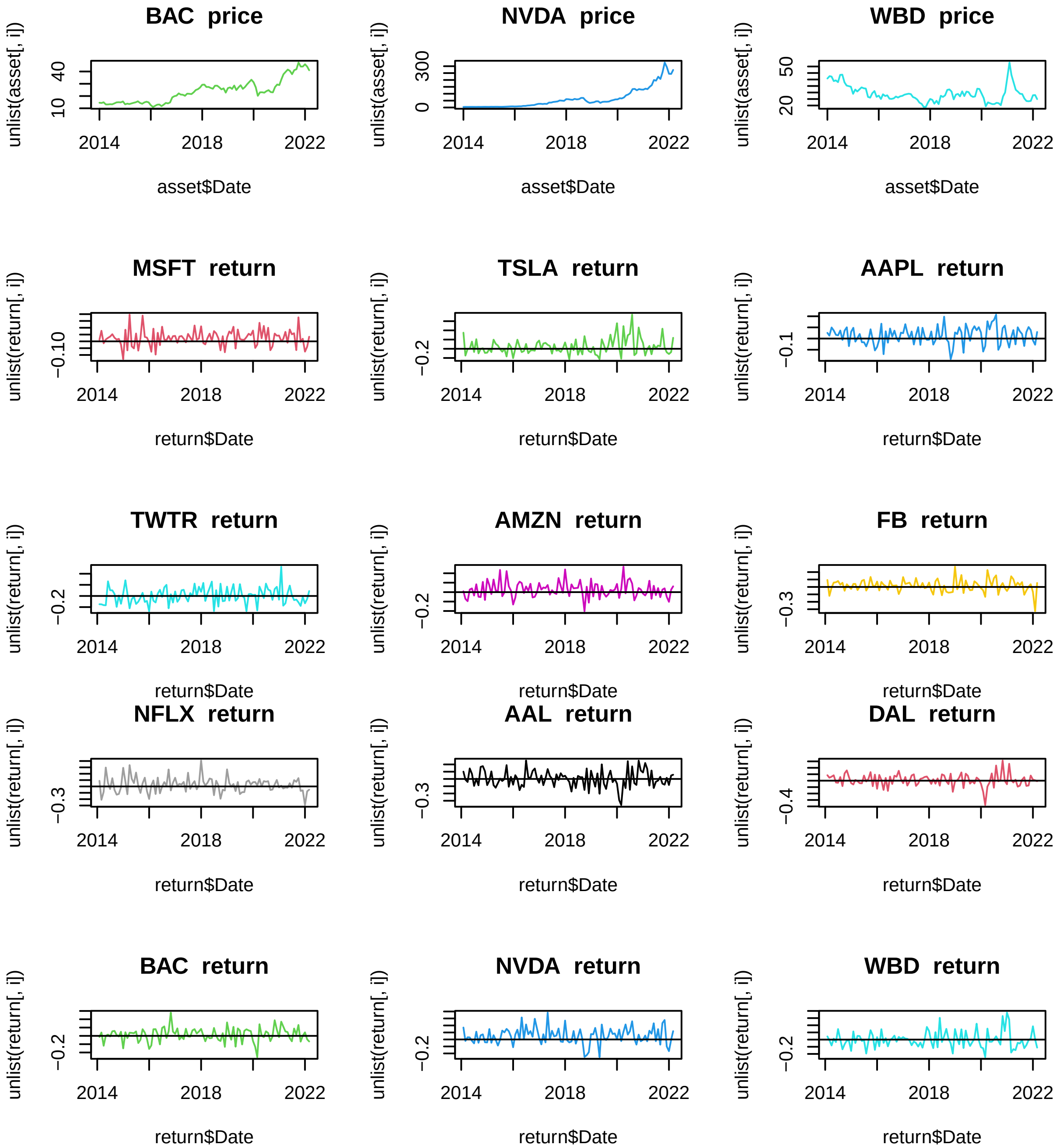
```
##       [,1]               [,2]               [,3]              [,4]
## names "MSFT"             "TSLA"             "AAPL"            "TWTR"
##       "2.62313579319408" "3.61192558378254" "2.8665143599088" "2.4884584653062"
##       [,5]               [,6]               [,7]
## names "AMZN"             "FB"               "NFLX"
##       "2.79378397789643" "2.83834274379434" "2.56064597670338"
##       [,8]               [,9]              [,10]
## names "AAL"              "DAL"             "BAC"
##       "3.17804029866847" "2.8024812841754" "3.05680636875636"
##       [,11]              [,12]
## names "NVDA"             "WBD"
##       "3.07147751026545" "2.86285142567178"
```

Plots

```
par(mfrow = c(3,3))
### Price
for(i in 2:13){
  plot(asset$Date, unlist(asset[,i]), type = "l", col = i,
       main = paste(colnames(asset[,i])," price"))
}
```

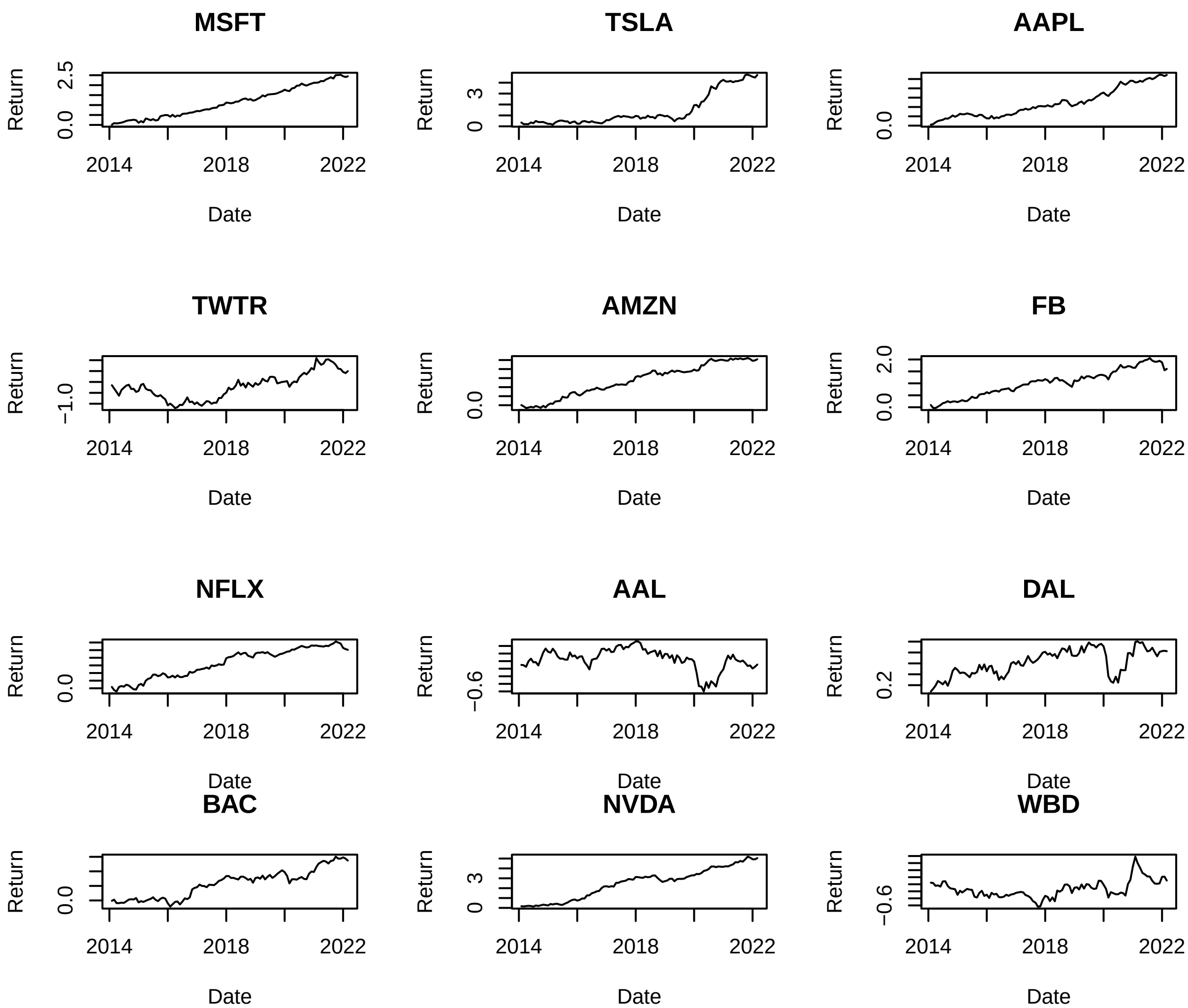



```r
## Return
for(i in 2:13){
  plot(return$Date, unlist(return[,i]), type = "l", col = i,
       main = paste(colnames(return[,i])," return"))
  abline(h=0)
}
```



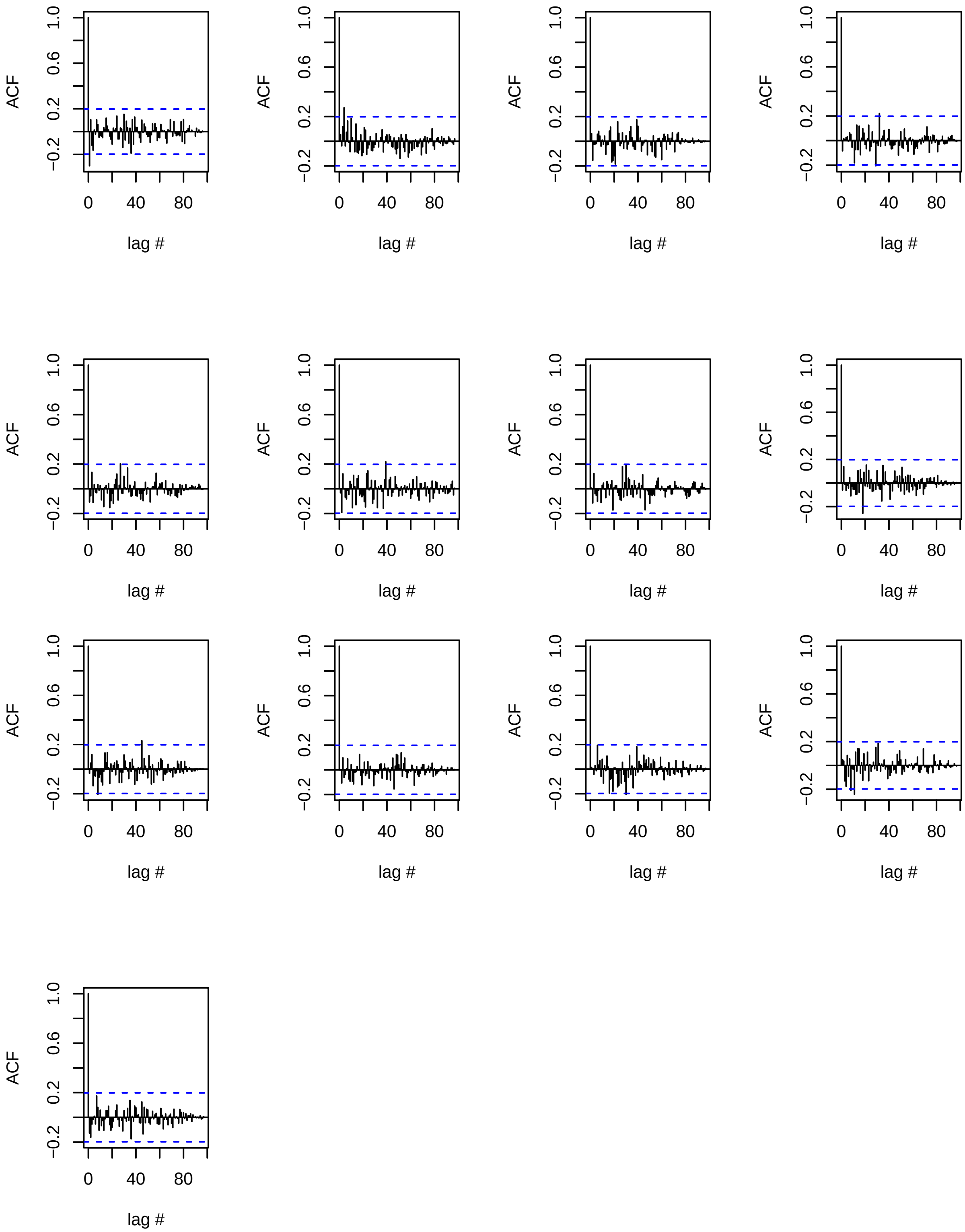
Equity curve

```
par(mfrow = c(3,3))
for(i in 2:13){
  plot(x = unlist(return$Date),y = unlist(cumsum(return[,i])), type="l",
       main = colnames(return[,i]), xlab = "Date", ylab = "Return")
}
```



Stationary Test

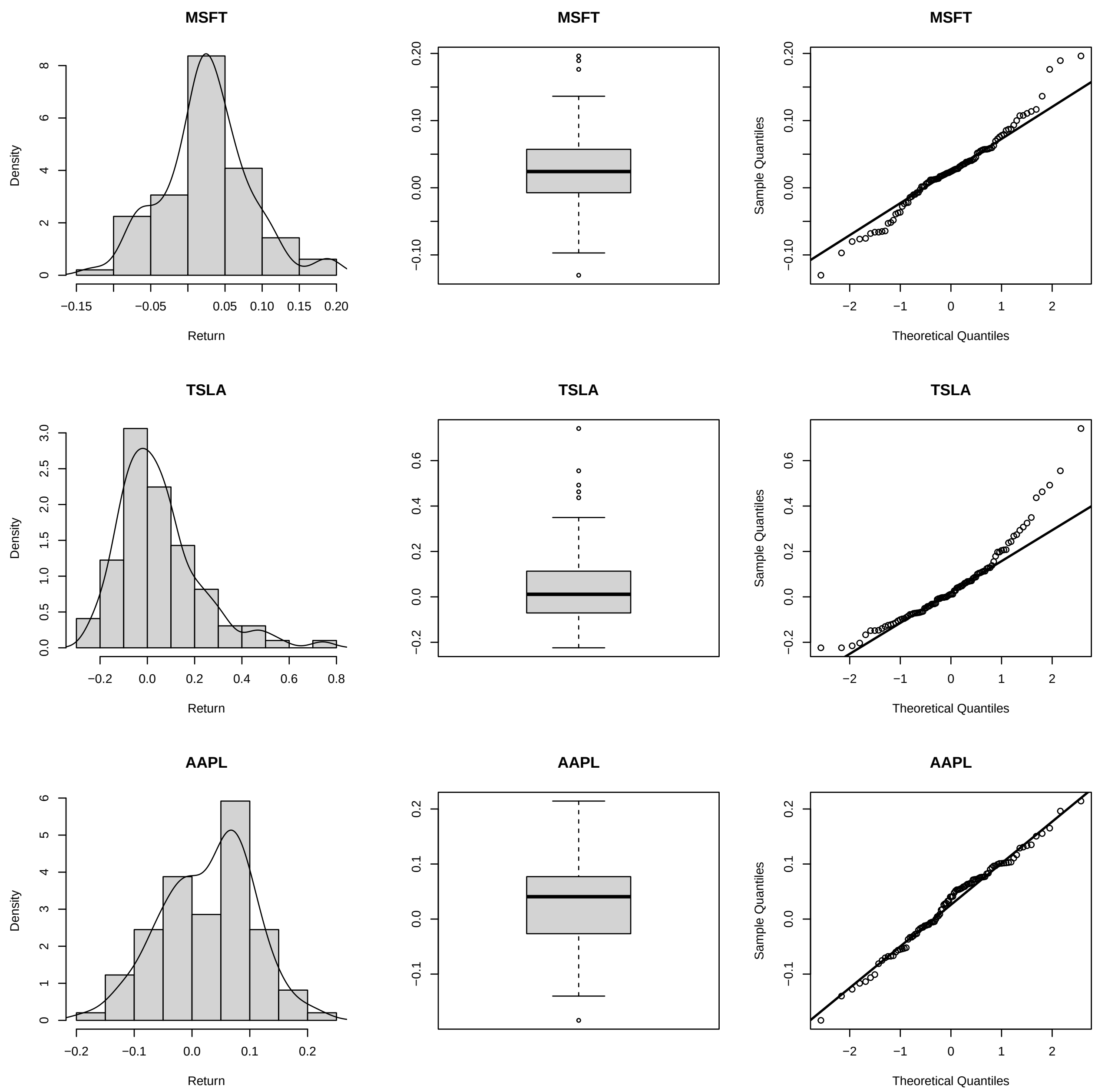
```
### Stationary Test
par(mfrow = c(2,4))
for(i in 2:14){
  acf(unlist(return[,i]),lag.max = length(return$MSFT),
         xlab = "lag #", ylab = 'ACF', main=' ')
}
```

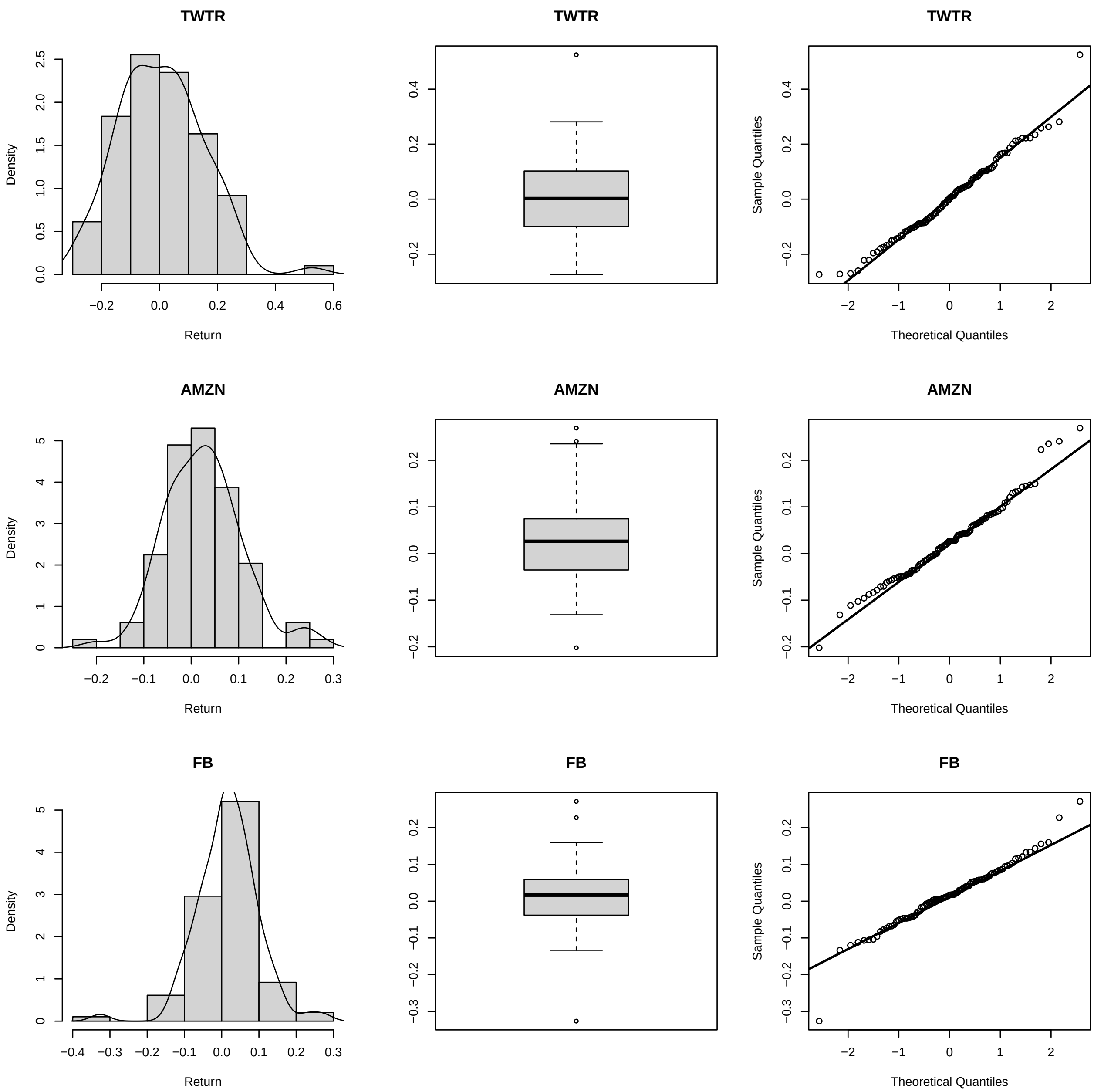


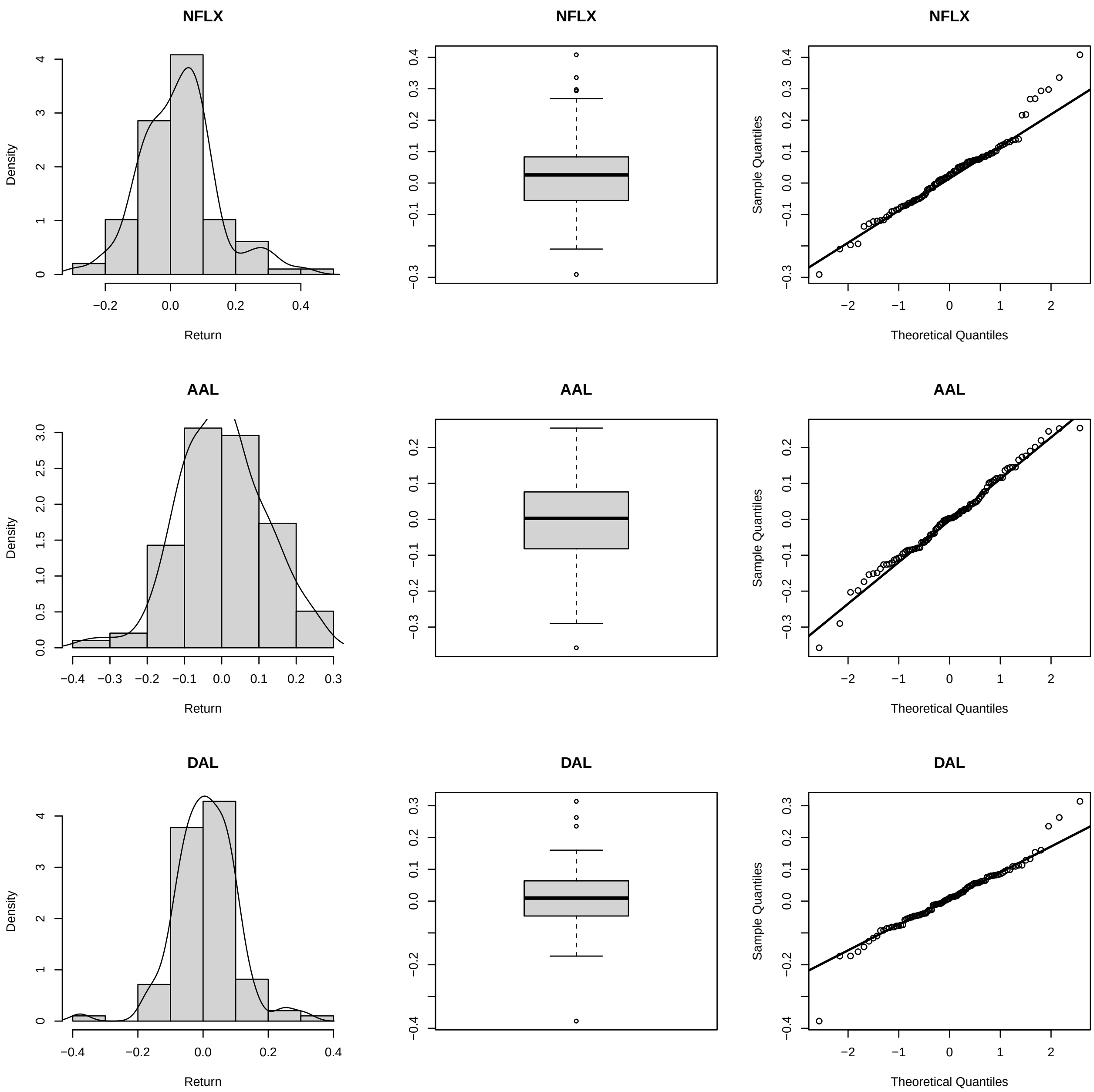
Hist, Boxplot, qqplot

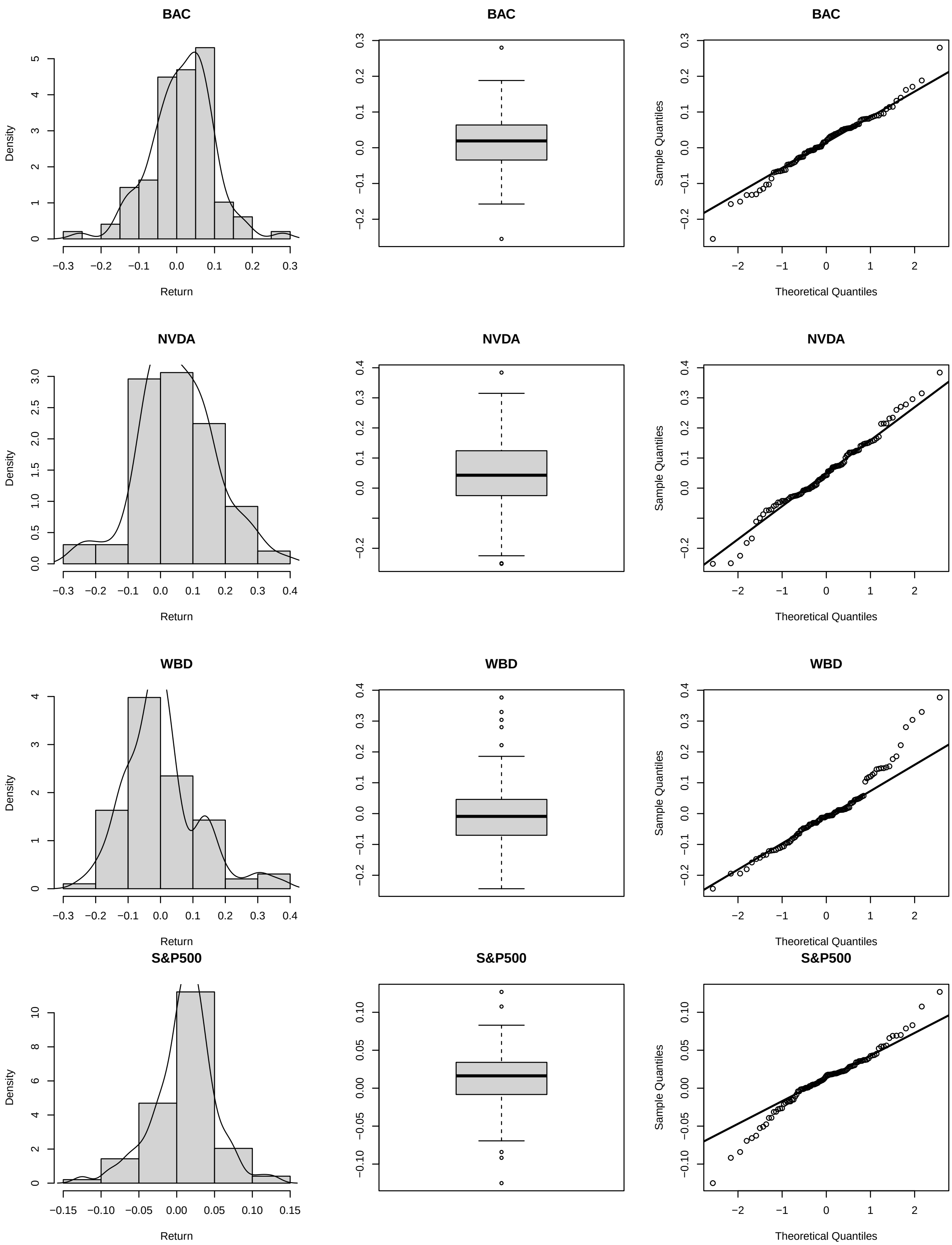
```r
### Hist, Boxplot, qqplot
par(mfrow = c(3,3))
```

```r
for(i in 2:14){
  hist(unlist(return[,i]), freq = FALSE,
       main = colnames(return)[i], xlab = "Return")
  lines(density(unlist(return[,i])))
  boxplot(unlist(return[,i]), main = colnames(return)[i])
  qqnorm(unlist(return[,i]), pch = 1, main = colnames(return)[i])
  qqline(unlist(return[,i]), lwd = 2)
}
```




TWTR
Density
Return
TWTR
TWTR
Sample Quantiles
Theoretical Quantiles
AMZN
Density
Return
AMZN
AMZN
Sample Quantiles
Theoretical Quantiles
FB
Density
Return
FB
FB
Sample Quantiles
Theoretical Quantiles


NFLX
Density
Return
NFLX
NFLX
Sample Quantiles
Theoretical Quantiles
AAL
Density
Return
AAL
AAL
Sample Quantiles
Theoretical Quantiles
DAL
Density
Return
DAL
DAL
Sample Quantiles
Theoretical Quantiles


BAC
Density
Return
BAC
BAC
Sample Quantiles
Theoretical Quantiles
NVDA
Density
Return
NVDA
NVDA
Sample Quantiles
Theoretical Quantiles
WBD
Density
Return
WBD
WBD
Sample Quantiles
Theoretical Quantiles
S&P500
Density
Return
S&P500
S&P500
Sample Quantiles
Theoretical Quantiles

Distributions

```
#### t
namesd <- data.frame(colnames(asset[1,2:13]))
tdis <- rep(NA, 12)
normal <- rep(NA, 12)
ged <- rep(NA, 12)

tdis_fun <- function(return) {
  start = c(mean(return), sd(return), 5)
  loglik_t = function(beta)
    sum(-dt((return - beta[1]) / beta[2],
            beta[3], log = TRUE) + log(beta[2]))
  fit_t = optim(
    start,
    loglik_t,
    hessian = T,
    method = "L-BFGS-B",
    lower = c(-1, 0.001, 1)
  )
  AIC_t = 2 * fit_t$value + 2 * 3
  #return(AIC_t)
  return(fit_t$value)
}

for (i in 2:13){
  tdis[i-1] <- lapply(return[,i], tdis_fun)
}
tdis <- data.frame(unlist(tdis))

#### normal
ndis_fun <- function(return) {
  AIC_n <- 2 * snormFit(return, hessian = TRUE)$objective + 2 * 3
  AIC_n
}
for(i in 2:13){
  normal[i-1] <- lapply(return[,i], ndis_fun)
}
normal <- data.frame(unlist(normal))

#### ged
ged_fun <- function(return) {
  AIC_ged <- 2 * gedFit(return, hessian = TRUE)$objective + 2 * 3
  AIC_ged
}
for(i in 2:13){
  ged[i-1] <- lapply(return[,i], ged_fun)
}
ged <- data.frame(unlist(ged))

dis_df <- cbind(namesd,tdis,normal,ged)
dis_df
```

```
##    colnames.asset.1..2.13.. unlist.tdis. unlist.normal. unlist.ged.
```

```
## 1                        MSFT   -139.66549     -271.25573  -276.23424
## 2                        TSLA    -38.58013      -80.34437   -69.20914
## 3                        AAPL   -112.15671     -220.30961  -218.90939
## 4                        TWTR    -50.63192      -97.29781   -94.90164
## 5                        AMZN   -107.95913     -209.86197  -209.57404
## 6                          FB   -111.98310     -209.20876  -217.11195
## 7                        NFLX    -73.72148     -138.40822  -140.89449
## 8                         AAL    -73.60630     -140.92136  -141.25665
## 9                         DAL    -96.76172     -178.09420  -184.85237
## 10                        BAC   -108.18352     -207.92323  -209.41384
## 11                       NVDA    -71.69179     -136.35263  -137.27907
## 12                        WBD    -77.72453     -151.06318  -153.27377
```

Sharpe's Slope ??

```r
# sharpes <- data.frame(matrix(ncol=13, nrow = 98))
# colnames(sharpes) <- colnames(return[1,2:14])
#
# for(i in 2:14){
#   sharpes[,i-1] = (unlist(return[,i])-unlist(return[,15])/100)/sds_r[i-1]
# }
# max(sharpes[,1])
names_sh <- data.frame(colnames(return[1,2:14]))
sharpes_list <- rep(NA, 13)
for(i in 2:14){
  sharpes_list[i-1] = (mean(unlist(return[,i]))-mean(unlist(return[,15]))/100)/sd(unlist(return[,i]))
}
sharpes_list <- data.frame(sharpes_list)
shar_df <- cbind(names_sh,sharpes_list)
shar_df
```

```
##    colnames.return.1..2.14.. sharpes_list
## 1                         MSFT   0.30124000
## 2                         TSLA   0.23644366
## 3                         AAPL   0.26624907
## 4                         TWTR  -0.01416282
## 5                         AMZN   0.23071594
## 6                           FB   0.11372558
## 7                         NFLX   0.15817121
## 8                          AAL  -0.05222523
## 9                          DAL   0.01341556
## 10                         BAC   0.08380025
## 11                        NVDA   0.37645499
## 12                         WBD  -0.05347243
## 13                      S&P500   0.08091924
```

M to Y

```r
means_y <- means_r*12
means_y
```

```
##       MSFT       TSLA       AAPL       TWTR       AMZN         FB       NFLX
## 0.29935838 0.57619081 0.33285316 0.06073613 0.31144214 0.19628869 0.30829647
##        AAL        DAL        BAC       NVDA        WBD     S&P500
```

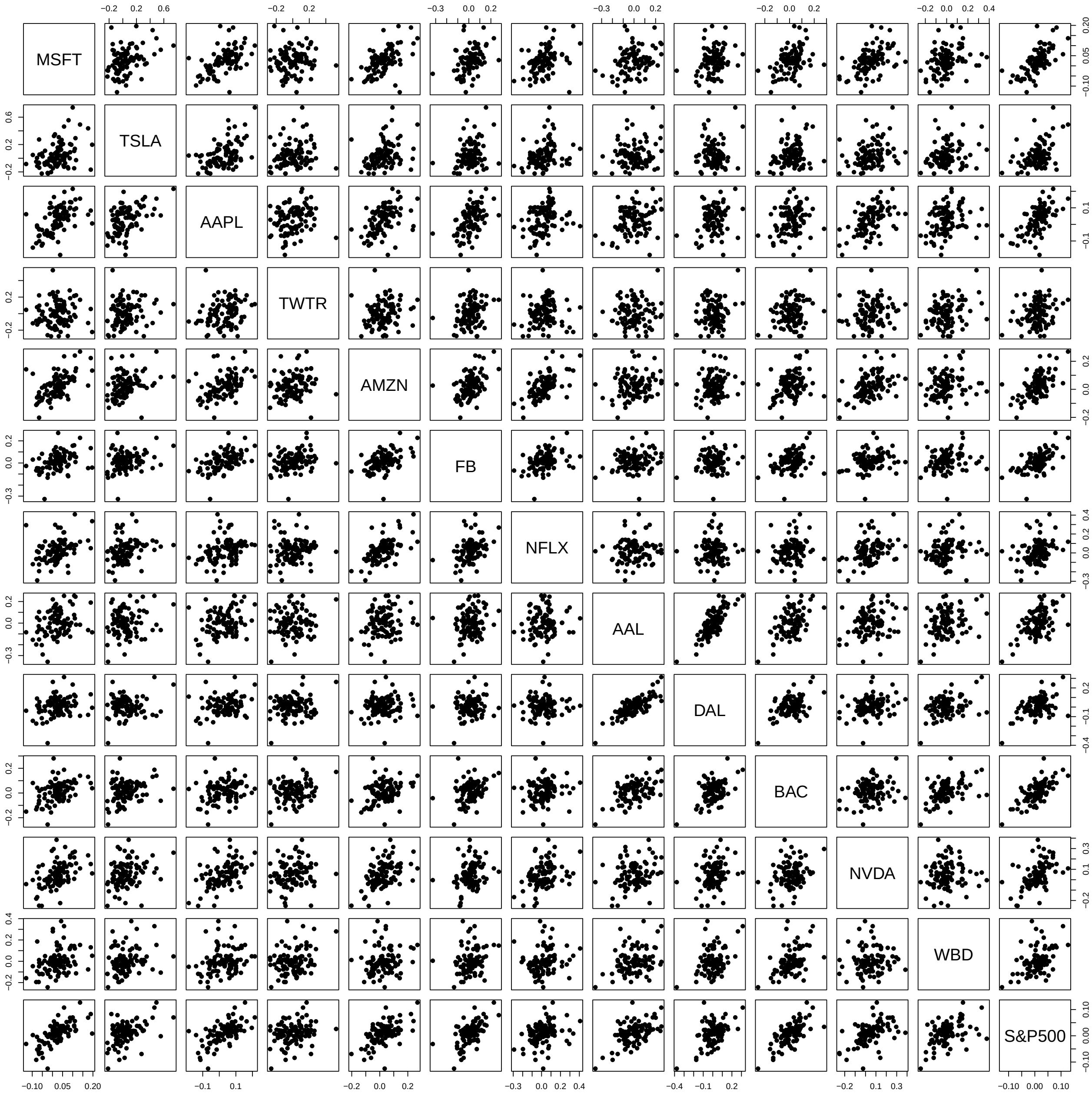
```
## 0.01341033 0.10074924 0.16781059 0.61751296 0.01272428 0.12430059
```

```
### SDs
sds_y <- means_r*sqrt(12)
sds_y
```

```
##        MSFT        TSLA        AAPL        TWTR        AMZN          FB
## 0.086417321 0.166331961 0.096086430 0.017533012 0.089905602 0.056663663
##        NFLX         AAL         DAL         BAC        NVDA         WBD
## 0.088997525 0.003871229 0.029083800 0.048442744 0.178260637 0.003673184
##      S&P500
## 0.035882490
```

Pairewise

```
pairs(return[,2:14],pch = 19)
```



Covariance Matrix

```
cov_mat <- cov(return[,2:14])
cov_mat
```

```
##                 MSFT        TSLA        AAPL         TWTR         AMZN
## MSFT    0.0035019726 0.003792370 0.002422600 0.0005161623 0.0026771843
## TSLA    0.0037923705 0.029916176 0.006264081 0.0035715448 0.0045237237
## AAPL    0.0024226004 0.006264081 0.005996676 0.0023691478 0.0028371440
## TWTR    0.0005161623 0.003571545 0.002369148 0.0211266367 0.0023404462
## AMZN    0.0026771843 0.004523724 0.002837144 0.0023404462 0.0066636395
## FB      0.0019051631 0.004147204 0.002966423 0.0029377213 0.0032040121
## NFLX    0.0025895014 0.005495064 0.002452145 0.0018368103 0.0055568361
## AAL     0.0020327162 0.003200935 0.002362101 0.0011806487 0.0014934263
```

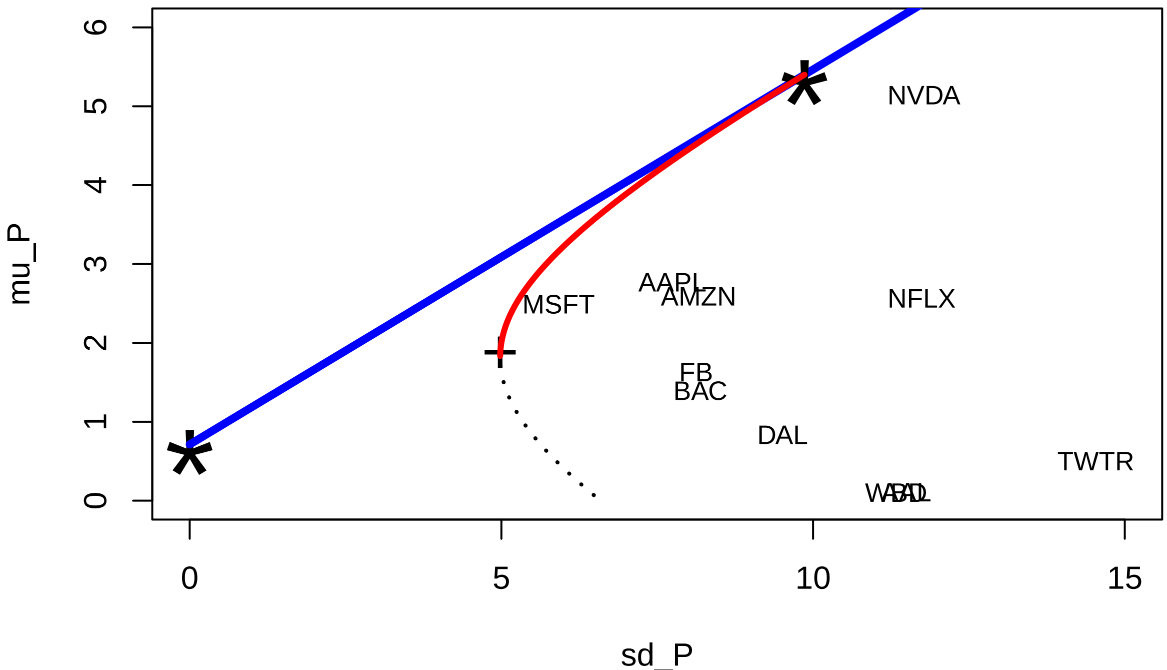
```
## DAL    0.0013915040 0.002040954 0.001857155 0.0023083974 0.0007854177
## BAC    0.0021169649 0.003438923 0.001598663 0.0023991294 0.0019667441
## NVDA   0.0033625952 0.005936788 0.004657367 0.0019210966 0.0042224020
## WBD    0.0019834923 0.004282038 0.001944331 0.0049875654 0.0017281794
## S&P500 0.0015817662 0.003165494 0.001971581 0.0013660558 0.0018550904
##                  FB         NFLX         AAL          DAL          BAC
## MSFT   0.0019051631 2.589501e-03 0.002032716 1.391504e-03 0.002116965
## TSLA   0.0041472040 5.495064e-03 0.003200935 2.040954e-03 0.003438923
## AAPL   0.0029664228 2.452145e-03 0.002362101 1.857155e-03 0.001598663
## TWTR   0.0029377213 1.836810e-03 0.001180649 2.308397e-03 0.002399129
## AMZN   0.0032040121 5.556836e-03 0.001493426 7.854177e-04 0.001966744
## FB     0.0065976698 3.566622e-03 0.001852641 8.328816e-04 0.002330298
## NFLX   0.0035666217 1.378597e-02 0.001261644 9.966518e-05 0.001838108
## AAL    0.0018526414 1.261644e-03 0.013209512 8.577703e-03 0.004995109
## DAL    0.0008328816 9.966518e-05 0.008577703 9.044550e-03 0.003932212
## BAC    0.0023302976 1.838108e-03 0.004995109 3.932212e-03 0.006709689
## NVDA   0.0027697463 5.112035e-03 0.003194241 2.367829e-03 0.002712336
## WBD    0.0020686813 2.086063e-03 0.003924688 3.583663e-03 0.004257161
## S&P500 0.0019264594 1.481677e-03 0.002470547 1.869021e-03 0.002276369
##                NVDA          WBD      S&P500
## MSFT   0.0033625952 0.0019834923 0.001581766
## TSLA   0.0059367881 0.0042820384 0.003165494
## AAPL   0.0046573671 0.0019443307 0.001971581
## TWTR   0.0019210966 0.0049875654 0.001366056
## AMZN   0.0042224020 0.0017281794 0.001855090
## FB     0.0027697463 0.0020686813 0.001926459
## NFLX   0.0051120350 0.0020860634 0.001481677
## AAL    0.0031942407 0.0039246877 0.002470547
## DAL    0.0023678293 0.0035836635 0.001869021
## BAC    0.0027123362 0.0042571614 0.002276369
## NVDA   0.0138725249 0.0007716842 0.002299867
## WBD    0.0007716842 0.0128416738 0.001965715
## S&P500 0.0022998673 0.0019657145 0.001601683
```

Portfolio Theory

With Short Sale

```r
library(quadprog)
R = 100*return[,2:13]
mean_p <- apply(R,2,mean)
cov_p <- cov(R)
sd_vect_p <- sqrt(diag(cov_p))
# min(mean_p)
# max(mean_p)
### With shortsale
M_p = length(mean_p)
Amat_p <- cbind(rep(1,M_p),mean_p)
mu_P = seq(0.07, 5.4, length = 300)
# Target portfolio means for the expect portfolio return
sd_P = mu_P # set up storage for std dev's of portfolio returns
weights_p = matrix(0, nrow = 300, ncol = M_p) # storage for return
for (i in 1:length(mu_P)) { # find the optimal portfolios
  bvec_p <- c(1, mu_P[i])
```

```r
  result_p = solve.QP(Dmat = 2 * cov_p, dvec = rep(0, M_p), Amat = Amat_p,
                      bvec = bvec_p, meq = 2)
  sd_P[i] = sqrt(result_p$value)
  weights_p[i, ] = result_p$solution
}
plot(sd_P, mu_P, type = "l", xlim = c(0,15), ylim = c(0, 6), lty = 3, lwd = 2)
# plot efficient frontier (and inefficient portfolios below the min var portfolio)
mufree_p = mean(return$`Treasury Bill 3 month (rf)`)# input value of risk-free interest rate
points(0, mufree_p, cex = 4, pch = "*") # show risk-free asset
sharpe_p = (mu_P - mufree_p) / sd_P # compute Sharpes ratios
ind_p = (sharpe_p == max(sharpe_p)) # Find maximum Sharpes ratio
#weights_p[ind_p,] # print the weights of the tangency portfolio
lines(c(0, 15), mufree_p + c(0, 15) * (mu_P[ind_p] - mufree_p) / sd_P[ind_p], lwd = 4,
      lty = 1, col = "blue") # show line of optimal portfolios
points(sd_P[ind_p], mu_P[ind_p], cex = 4, pch = "*") # tangency portfolio
ind2_p = (sd_P == min(sd_P)) # find the minimum variance portfolio
points(sd_P[ind2_p], mu_P[ind2_p], cex = 2, pch = "+") # min var portfolio
ind3_p = (mu_P > mu_P[ind2_p])
lines(sd_P[ind3_p], mu_P[ind3_p], type = "l", xlim = c(0, 25), ylim = c(0,30),
      lwd = 3, col = 'red') # plot the efficient frontier
for(i in 1:12){
  text(sd_vect_p[i], mean_p[i],colnames(return[,i+1]), cex=0.8)
}
```




```r
### MVP
(mvp_meanreturn <- mu_P[ind2_p])
```

```
## [1] 1.816957
```

```r
(mvp_sd <- sd_P[ind2_p])
```

```
## [1] 4.980327
```

```r
weights_mvp <- weights_p[ind2_p,]
weights_mvp <- t(data.frame(weights_mvp))
colnames(weights_mvp) <- colnames(return[2:13])
weights_mvp
```

```
##                  MSFT        TSLA      AAPL       TWTR       AMZN        FB
## weights_mvp 0.5017466 -0.04977035 0.1347946 0.04885673 0.03852826 0.1315965
##                  NFLX         AAL       DAL        BAC        NVDA         WBD
## weights_mvp 0.0430567 -0.09160249 0.2182182 0.08071435 -0.05955131 0.003412278
```

```r
(mvp_meanreturn_ann <- mvp_meanreturn*12)
```

```
## [1] 21.80348
```

```r
(mvp_sd_ann <- mvp_sd*sqrt(12))
```

```
## [1] 17.25236
```

```r
### Efficient Portfolio Frontier
EPF_mean <- mu_P[ind3_p]
EPF_sd <- sd_P[ind3_p]

### Tangency Portfolio
(tan_meanreturn <- mu_P[ind_p])
```

```
## [1] 5.4
```

```r
(tan_sd <- sd_P[ind_p])
```

```
## [1] 9.86121
```

```r
(tan_var <- tan_sd^2)
```

```
## [1] 97.24347
```

```r
(tan_sharpes <- (tan_meanreturn-mufree_p)/tan_sd)
```

```
## [1] 0.4753989
```

```r

### Tail dependence can be seen among the assets, therefore we can fit
### our portfolio with multivariate t-distribution.

## MVP VaR&ES
library(MASS)
alpha = 0.05
return_mvp <- rowSums(data.frame(
  weights_mvp[1] * return[, 2],
  weights_mvp[2] * return[, 3],
  weights_mvp[3] * return[, 4],
  weights_mvp[4] * return[, 5],
  weights_mvp[5] * return[, 6],
  weights_mvp[6] * return[, 7],
  weights_mvp[7] * return[, 8],
  weights_mvp[8] * return[, 9],
  weights_mvp[9] * return[, 10],
  weights_mvp[10] * return[, 11],
  weights_mvp[11] * return[, 12],
```

```
  weights_mvp[12] * return[, 13]))
fitt_mvp = fitdistr(return_mvp,"t")
param_mvp = as.numeric(fitt_mvp$estimate)
mean_mvpfit = param_mvp[1]
df_mvpfit = param_mvp[3]
sd_mvpfit = param_mvp[2] * sqrt((df_mvpfit) / (df_mvpfit - 2))
lambda_mvpfit = param_mvp[2]
qalpha_mvp = qt(alpha, df = df_mvpfit)
VaR_par_mvp = -100000 * (mean_mvpfit + lambda_mvpfit * qalpha_mvp)
es1_mvp = dt(qalpha_mvp, df = df_mvpfit) / (alpha)
es2_mvp=(df_mvpfit+qalpha_mvp^2)/(df_mvpfit-1)
es3_mvp=-mean_mvpfit+lambda_mvpfit*es1_mvp*es2_mvp
ES_par_mvp = 100000*es3_mvp
VaR_par_mvp
```

VaR&ES

```
## [1] 6285.691
```

```
ES_par_mvp
```

```
## [1] 8957.941
```

```
## Asset VaR
S0 = 100000
qnalpha = qnorm(0.05)

### MSFT
q_msft = as.numeric(quantile(return$MSFT, alpha))
VAR_msft = -S0 * q_msft
#VAR_msft

### TSLA
fit_tsla <- fitdistr(return$TSLA, "normal")
param_tsla = as.numeric(fit_tsla$estimate)
mean_tsla = param_tsla[1]
sd_tsla = param_tsla[2]
VAR_tsla = -S0*(mean_tsla+qnalpha*sd_tsla)
#VAR_tsla

### AAPL
fit_aapl <- fitdistr(return$AAPL, "normal")
param_aapl = as.numeric(fit_aapl$estimate)
mean_aapl = param_aapl[1]
sd_aapl = param_aapl[2]
VAR_aapl = -S0*(mean_aapl+qnalpha*sd_aapl)
#VAR_aapl

### TWTR
fit_twtr <- fitdistr(return$TWTR, "normal")
param_twtr = as.numeric(fit_twtr$estimate)
mean_twtr = param_twtr[1]
sd_twtr = param_twtr[2]
VAR_twtr = -S0*(mean_twtr+qnalpha*sd_twtr)
#VAR_twtr
```

```r
### AMZN
fit_amzn <- fitdistr(return$AMZN, "normal")
param_amzn = as.numeric(fit_amzn$estimate)
mean_amzn = param_amzn[1]
sd_amzn = param_amzn[2]
VAR_amzn = -S0*(mean_amzn+qnalpha*sd_amzn)
#VAR_amzn

### FB
q_fb = as.numeric(quantile(return$FB, alpha))
VAR_fb = -S0 * q_fb
#VAR_fb

### NFLX
q_nflx = as.numeric(quantile(return$NFLX, alpha))
VAR_nflx = -S0 * q_nflx
#VAR_nflx

### AAL
q_aal = as.numeric(quantile(return$AAL, alpha))
VAR_aal = -S0 * q_aal
#VAR_aal

### DAL
q_dal = as.numeric(quantile(return$DAL, alpha))
VAR_dal = -S0 * q_dal
#VAR_dal

### BAC
q_bac = as.numeric(quantile(return$BAC, alpha))
VAR_bac = -S0 * q_bac
#VAR_bac

### NVDA
q_nvda = as.numeric(quantile(return$NVDA, alpha))
VAR_nvda = -S0 * q_nvda
#VAR_nvda

### WBD
q_wbd = as.numeric(quantile(return$WBD, alpha))
VAR_wbd = -S0 * q_wbd
#VAR_wbd

VAR_asset <- c(VAR_msft, VAR_tsla, VAR_aapl, VAR_twtr, VAR_amzn, VAR_fb, VAR_nflx,
               VAR_aal, VAR_dal, VAR_bac, VAR_nvda, VAR_wbd)
cbind(names, VAR_asset)
```

```
##       names  VAR_asset
##  [1,] "MSFT" "6918.17869693084"
##  [2,] "TSLA" "23502.7562034031"
##  [3,] "AAPL" "9898.52221650451"
##  [4,] "TWTR" "23279.5289061133"
##  [5,] "AMZN" "10763.0915501661"
```

```
##  [6,] "FB"   "10561.0446590248"
##  [7,] "NFLX" "13072.710950181"
##  [8,] "AAL"  "15706.0583956349"
##  [9,] "DAL"  "12891.9240002285"
## [10,] "BAC"  "13041.1530377474"
## [11,] "NVDA" "11974.7969532429"
## [12,] "WBD"  "14903.9099722124"
```

```
names_sh <- data.frame(colnames(return[1,2:14]))
sharpes_list <- rep(NA, 13)
for(i in 2:14){
  sharpes_list[i-1] = (mean(unlist(return[,i]))-mean(unlist(return[,15]))/100)/sd(unlist(return[,i]))
}
sharpes_list <- data.frame(sharpes_list)
shar_df <- cbind(names_sh,sharpes_list)
shar_df
```

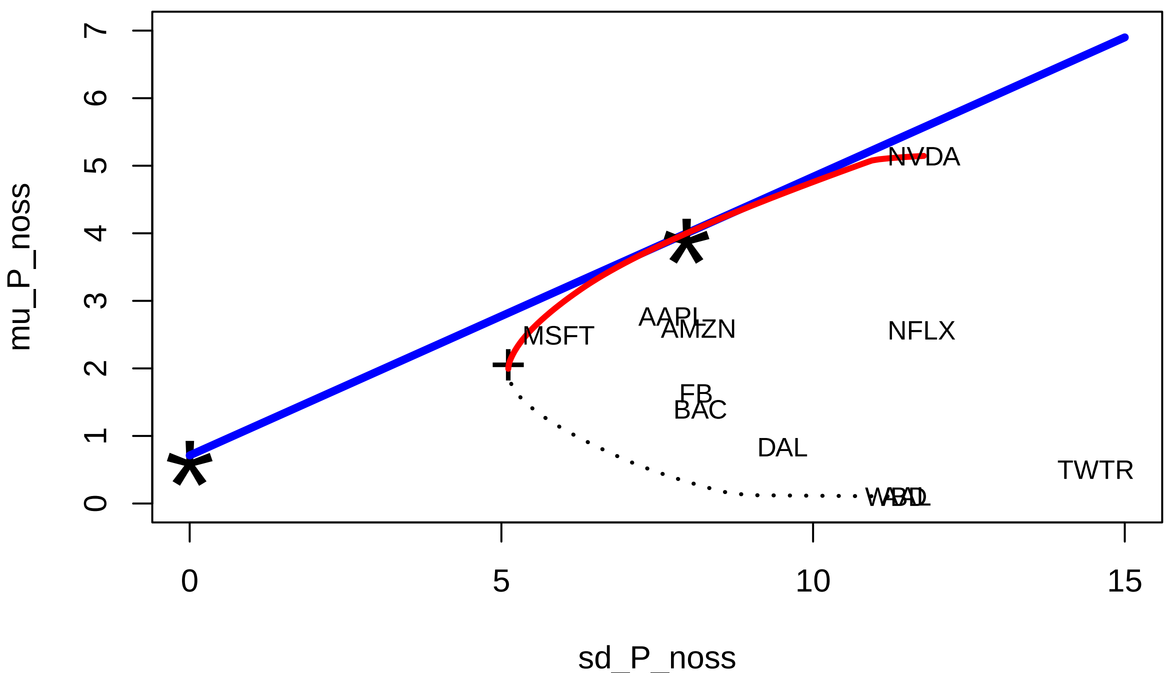
Assets' Sharpe's Ratios

```
##    colnames.return.1..2.14.. sharpes_list
## 1                       MSFT   0.30124000
## 2                       TSLA   0.23644366
## 3                       AAPL   0.26624907
## 4                       TWTR  -0.01416282
## 5                       AMZN   0.23071594
## 6                         FB   0.11372558
## 7                       NFLX   0.15817121
## 8                        AAL  -0.05222523
## 9                        DAL   0.01341556
## 10                       BAC   0.08380025
## 11                      NVDA   0.37645499
## 12                       WBD  -0.05347243
## 13                    S&P500   0.08091924
```

Without shortshale

```
### Without shortsale
#R = 100*return[,2:13]
#mean_p <- apply(R,2,mean)
#cov_p <- cov(R)
#sd_vect_p <- sqrt(diag(cov_p))
### With shortsale
#M_p = length(mean_p)
Amat_p_noss <- cbind(rep(1,M_p),mean_p, diag(1,nrow=M_p))
mu_P_noss = seq(min(mean_p)+0.0001, max(mean_p)-0.0001, length = 300)
# Target portfolio means for the expect portfolio return
sd_P_noss = mu_P_noss # set up storage for std dev's of portfolio returns
weights_p_noss = matrix(0, nrow = 300, ncol = M_p) # storage for return
for (i in 1:length(mu_P_noss)) { # find the optimal portfolios
  bvec_p_noss <- c(1, mu_P_noss[i], rep(0,M_p))
  result_noss = solve.QP(Dmat = 2 * cov_p, dvec = rep(0, M_p), Amat = Amat_p_noss,
                   bvec = bvec_p_noss, meq = 2)
  sd_P_noss[i] = sqrt(result_noss$value)
```

```r
  weights_p_noss[i, ] = result_noss$solution
}
plot(sd_P_noss, mu_P_noss, type = "l", lty = 3,
     lwd = 2, xlim = c(0,15), ylim = c(0,7))
# plot efficient frontier (and inefficient portfolios below the min var portfolio)
#mufree_p = mean(return$`Treasury Bill 3 month (rf)`) # input value of risk-free interest rate
points(0, mufree_p, cex = 4, pch = "*") # show risk-free asset
sharpe_p_noss = (mu_P_noss - mufree_p) / sd_P_noss # compute Sharpes ratios
ind_p_noss = (sharpe_p_noss == max(sharpe_p_noss)) # Find maximum Sharpes ratio
#weights_p[ind_p,] # print the weights of the tangency portfolio
lines(c(0, 15), mufree_p + c(0, 15) * (mu_P_noss[ind_p_noss] - mufree_p) / sd_P_noss[ind_p_noss], lwd =
      lty = 1, col = "blue") # show line of optimal portfolios
points(sd_P_noss[ind_p_noss], mu_P_noss[ind_p_noss], cex = 4, pch = "*") # tangency portfolio
ind2_p_noss = (sd_P_noss == min(sd_P_noss)) # find the minimum variance portfolio
points(sd_P_noss[ind2_p_noss], mu_P_noss[ind2_p_noss], cex = 2, pch = "+") # min var portfolio
ind3_p_noss = (mu_P_noss > mu_P_noss[ind2_p_noss])
lines(sd_P_noss[ind3_p_noss], mu_P_noss[ind3_p_noss], type = "l",
      lwd = 3, col = 'red') # plot the efficient frontier
for(i in 1:12){
  text(sd_vect_p[i], mean_p[i],colnames(return[,i+1]), cex=0.8)
}
```




```r
### MVP
(mvp_meanreturn_noss <- mu_P_noss[ind2_p_noss])
```

```
## [1] 1.977063
```

```r
(mvp_sd_noss <- sd_P_noss[ind2_p_noss])
```

```
## [1] 5.110456
```

```r
weights_mvp_noss <- weights_p_noss[ind2_p_noss,]
weights_mvp_noss <- t(data.frame(weights_mvp_noss))
colnames(weights_mvp_noss) <- colnames(return[2:13])
```

```r
weights_mvp_noss
```

```
##                       MSFT          TSLA       AAPL       TWTR       AMZN
## weights_mvp_noss 0.487966 -1.764314e-18 0.06592605 0.05609903 0.03247623
##                         FB       NFLX           AAL      DAL        BAC
## weights_mvp_noss 0.1346882 0.02256666 -2.931383e-17 0.146045 0.04453321
##                          NVDA         WBD
## weights_mvp_noss 6.441445e-18 0.009699598
```

```r
(mvp_meanreturn_ann_noss <- mvp_meanreturn_noss*12)
```

```
## [1] 23.72476
```

```r
(mvp_sd_ann_noss <- mvp_sd_noss*sqrt(12))
```

```
## [1] 17.70314
```

```r
### Efficient Portfolio Frontier
EPF_mean_noss <- mu_P_noss[ind3_p_noss]
EPF_sd_noss <- sd_P_noss[ind3_p_noss]

### Tangency Portfolio
(tan_meanreturn_noss <- mu_P_noss[ind_p_noss])
```

```
## [1] 3.999688
```

```r
(tan_sd_noss <- sd_P_noss[ind_p_noss])
```

```
## [1] 7.96967
```

```r
(tan_var_noss <- tan_sd_noss^2)
```

```
## [1] 63.51564
```

```r
(tan_sharpes_noss <- (tan_meanreturn_noss-mufree_p)/tan_sd_noss)
```

```
## [1] 0.412526
```