Comparison of Returns using CAPM and Real world

Initial Setup

```
library(dplyr)
library(ggplot2)
library(lubridate)
library(quantmod)
library(xts)
library(psych)
library(IntroCompFinR)
library("readxl")
library(kableExtra)

MSFT <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "MSFT")
MSFT$Date <- as.Date(as.character(MSFT$Date))
MSFT = subset(MSFT, select = -c(Volume, Open, High, Low, Close))
MSFT$AdjClose <- as.numeric(MSFT$AdjClose)
MSFT <- xts(MSFT$AdjClose, order.by = MSFT$Date)
INTC <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "INTC")
INTC$Date <- as.Date(as.character(INTC$Date))
INTC = subset(INTC, select = -c(Volume, Open, High, Low, Close))
INTC$AdjClose <- as.numeric(INTC$AdjClose)
INTC <- xts(INTC$AdjClose, order.by = INTC$Date)
LUV <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "LUV")
LUV$Date <- as.Date(as.character(LUV$Date))
LUV = subset(LUV, select = -c(Volume, Open, High, Low, Close))
LUV$AdjClose <- as.numeric(LUV$AdjClose)
LUV <- xts(LUV$AdjClose, order.by = LUV$Date)
MCD <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "MCD")
MCD$Date <- as.Date(as.character(MCD$Date))
MCD = subset(MCD, select = -c(Volume, Open, High, Low, Close))
MCD$AdjClose <- as.numeric(MCD$AdjClose)
MCD <- xts(MCD$AdjClose, order.by = MCD$Date)
JNJ <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "JNJ")
JNJ$Date <- as.Date(as.character(JNJ$Date))
JNJ = subset(JNJ, select = -c(Volume, Open, High, Low, Close))
JNJ$AdjClose <- as.numeric(JNJ$AdjClose)
JNJ <- xts(JNJ$AdjClose, order.by = JNJ$Date)
RF <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                   sheet = "F-F_Research_Data_Factors_daily")
RF$Date <- as.Date(as.character(RF$Date))
RF = subset(RF, select = -c(SMB, HML, rawdate))
```

```
RF$RF <- as.numeric(RF$RF)
RF_p <- 0
for(i in 1:length(RF$RF)) {
  if(i==1) {RF_p[i] <- (1+RF$RF[i]/100)}
  else{RF_p[i] <- RF_p[i-1]*(1+RF$RF[i]/100)}
}
RF <- xts(RF_p, order.by = RF$Date)

MktRF <- read_excel('~/Documents/Documents/Investment Codes/lecture6p.xlsx',
                    sheet ="F-F_Research_Data_Factors_daily")
MktRF$Date <- as.Date(as.character(MktRF$Date))
MktRF = subset(MktRF, select = -c(SMB, HML, rawdate))
MktRF$MktRF <- as.numeric(MktRF$MktRF)
MktRF_p <- 0
for(i in 1:length(MktRF$MktRF)) {
  if(i==1) {MktRF_p[i] <- (1+MktRF$MktRF[i]/100)}
  else{MktRF_p[i] <- MktRF_p[i-1]*(1+MktRF$MktRF[i]/100)}
  }
MktRF <- xts(MktRF_p, order.by = MktRF$Date)
nss <- function(t, b0, b1, b2, b3, t1, t2) {
    r <- b0 + b1*((1-exp(-t/t1))/(t/t1)) + (b2*(((1-exp(-t/t1))/(t/t1))-
          (exp(-t/t1)))) +  (b3*(((1-exp(-t/t2))/(t/t2))-(exp(-t/t2))))
    return(r)
}

calculate_bond_price <- function(maturity, par, rate) {
  bond_price <- par/((1+(rate/100))^maturity)
  return(bond_price)
}
```

(1) Construct weekly simple total returns from the price data (use Adj Close to include div- idends). Compute and report the weekly and annualized mean and standard deviation for each stock. Compute the correlation matrix.

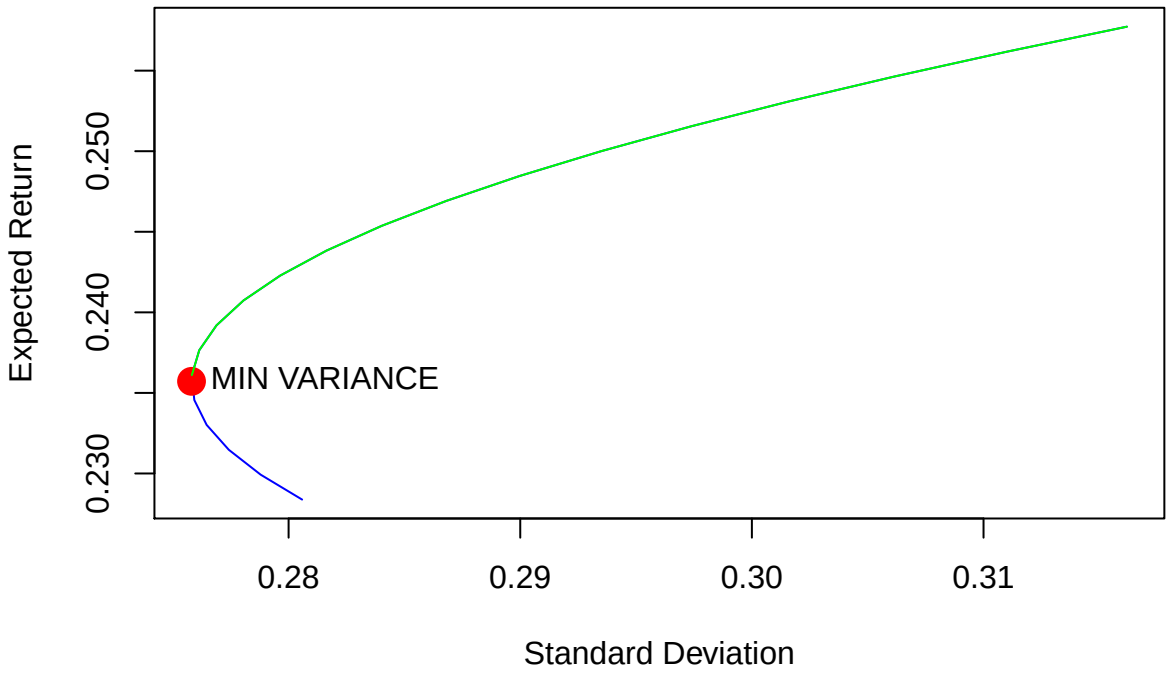
```
MSFT_return <- weeklyReturn(MSFT)
colnames(MSFT_return) <- "MSFT"
MSFT_sd <- sd(MSFT_return)*sqrt(52)
MSFT_mean <- mean(MSFT_return) * 52
INTC_return <- weeklyReturn(INTC)
colnames(INTC_return) <- "INTC"
INTC_sd <- sd(INTC_return)*sqrt(52)
INTC_mean <- mean(INTC_return) * 52
LUV_return <- weeklyReturn(LUV)
colnames(LUV_return) <- "LUV"
LUV_sd <- sd(LUV_return)*sqrt(52)
LUV_mean <- mean(LUV_return) * 52
MCD_return <- weeklyReturn(MCD)
colnames(MCD_return) <- "MCD"
MCD_sd <- sd(MCD_return)*sqrt(52)
MCD_mean <- mean(MCD_return) * 52
JNJ_return <- weeklyReturn(JNJ)
colnames(JNJ_return) <- "JNJ"
JNJ_sd <- sd(JNJ_return)*sqrt(52)
JNJ_mean <- mean(JNJ_return) * 52
```

```
RF_return <- weeklyReturn(RF)
colnames(RF_return) <- "RF"
RF_sd <- sd(RF_return)*sqrt(52)
RF_mean <- mean(RF_return) * 52
MktRF_return <- weeklyReturn(MktRF)
colnames(MktRF_return) <- "MktRF"
MktRF_sd <- sd(MktRF_return)*sqrt(52)
MktRF_mean <- mean(MktRF_return) * 52
Corr_matrix <- cor(data.frame(MSFT_return, INTC_return, LUV_return,
                              MCD_return, JNJ_return))
Cov_All <- cov(data.frame(MSFT_return, INTC_return, LUV_return,
                              MCD_return, JNJ_return)) * 52
Cov_MSFT_INTC <- cov(data.frame(MSFT_return, INTC_return)) * 52

data.frame("Stock" = c("MSFT", "INTC", "LUV", "MCD", "JNJ"),
           "Mean" = c(MSFT_mean, INTC_mean, LUV_mean, MCD_mean, JNJ_mean),
           "Standard Deviation" =c(MSFT_sd, INTC_sd, LUV_sd, MCD_sd, JNJ_sd))%>%
  kbl() %>%kable_styling()
```

Stock	Mean	Standard.Deviation
MSFT	0.2503915	0.2944417
INTC	0.2040150	0.3544914
LUV	0.1789139	0.3430972
MCD	0.1491664	0.2256500
JNJ	0.1423391	0.2050854

(2) Construct the mean-variance frontier for the Intel-Microsoft combination. Indicate the minimum-variance portfolio and the efficient frontier (the efficient frontier is a set of expected returns - risks that you would want to consider investing in).

```
ef <- efficient.frontier(c(MSFT_mean,INTC_mean), Cov_MSFT_INTC)

# compute global minimum variance portfolio for Microsoft and Intel

gmin.port = globalMin.portfolio(c(MSFT_mean,INTC_mean), Cov_MSFT_INTC)
plot(ef$sd, ef$er, col="blue", pch=16, type = 'l', xlab = "Standard Deviation",
     ylab = "Expected Return")
points(gmin.port$sd, gmin.port$er, col="red", pch=16, cex=2)
text(gmin.port$sd, gmin.port$er, labels="MIN VARIANCE", pos=4)
lines(ef$sd[1:15], ef$er[1:15], col="green", pch=16, type = 'l')
```




```
ef
```

```
## Call:
## efficient.frontier(er = c(MSFT_mean, INTC_mean), cov.mat = Cov_MSFT_INTC)
##
## Frontier portfolios' expected returns and standard deviations
##    port 1 port 2 port 3 port 4 port 5 port 6 port 7 port 8 port 9 port 10
## ER 0.2577 0.2562 0.2546 0.2531 0.2515 0.2500 0.2485 0.2469 0.2454  0.2438
## SD 0.3162 0.3110 0.3062 0.3016 0.2974 0.2935 0.2900 0.2868 0.2840  0.2816
##    port 11 port 12 port 13 port 14 port 15 port 16 port 17 port 18 port 19
## ER  0.2423  0.2407  0.2392  0.2376  0.2361  0.2346  0.2330  0.2315  0.2299
## SD  0.2796  0.2781  0.2769  0.2761  0.2758  0.2759  0.2765  0.2774  0.2788
##    port 20
## ER  0.2284
## SD  0.2806
```

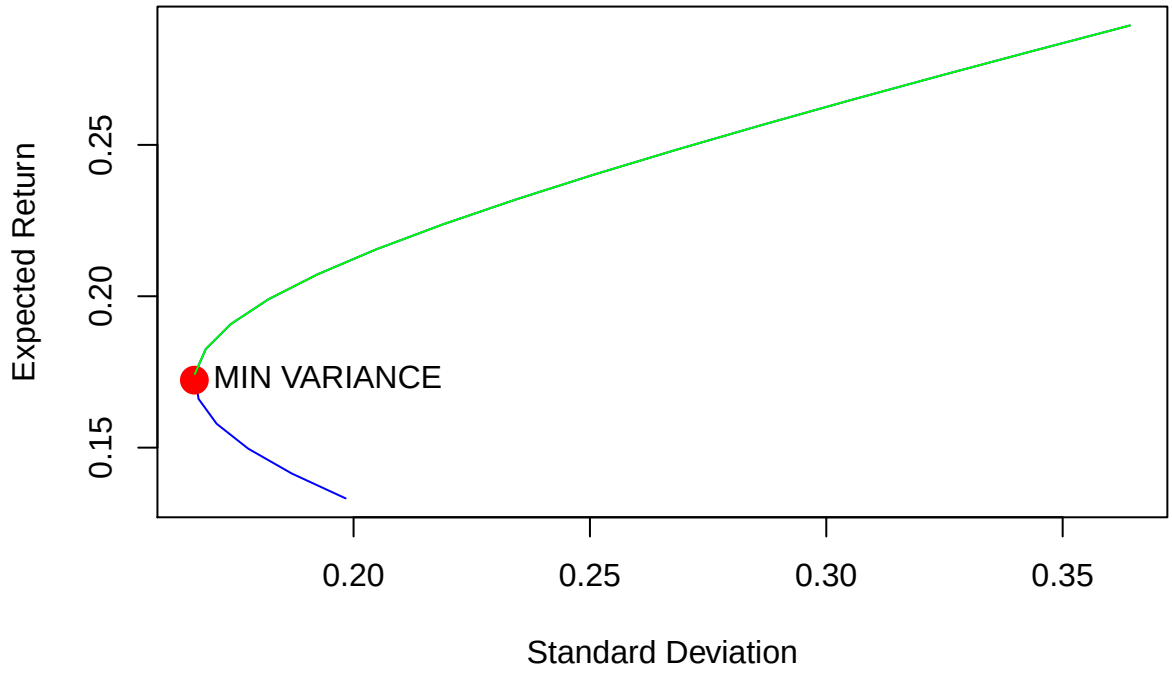
```
gmin.port
```

```
## Call:
## globalMin.portfolio(er = c(MSFT_mean, INTC_mean), cov.mat = Cov_MSFT_INTC)
##
## Portfolio expected return:     0.2357174
## Portfolio standard deviation:  0.2758093
## Portfolio weights:
## [1] 0.6836 0.3164
```

3. Add remaining stocks to the mix. Compute the mean-variance frontier and plot it on the same chart with the one from the previous question. Indicate the minimum variance portfolio and the efficient frontier. How do they compare to those of the previous question?

```
ef_all <- efficient.frontier(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean, MCD_mean)
                             , Cov_All)

# compute global minimum variance portfolio for all assets
```

```
gmin.port.all = globalMin.portfolio(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean,
                                      MCD_mean), Cov_All)
plot(ef_all$sd, ef_all$er, col="blue", pch=16, type = 'l',
     xlab = "Standard Deviation", ylab = "Expected Return")
points(gmin.port.all$sd, gmin.port.all $er, col="red", pch=16, cex=2)
text(gmin.port.all$sd, gmin.port.all$er, labels="MIN VARIANCE", pos=4)
lines(ef_all$sd[1:15], ef_all$er[1:15], col="green", pch=16, type = 'l')
```




```
ef_all
```

```
## Call:
## efficient.frontier(er = c(MSFT_mean, INTC_mean, JNJ_mean, LUV_mean,
##     MCD_mean), cov.mat = Cov_All)
##
## Frontier portfolios' expected returns and standard deviations
##    port 1 port 2 port 3 port 4 port 5 port 6 port 7 port 8 port 9 port 10
## ER 0.2894 0.2812 0.2730 0.2648 0.2566 0.2483 0.2401 0.2319 0.2237  0.2155
## SD 0.3642 0.3442 0.3244 0.3051 0.2863 0.2682 0.2507 0.2342 0.2188  0.2047
##    port 11 port 12 port 13 port 14 port 15 port 16 port 17 port 18 port 19
## ER  0.2072   0.199  0.1908  0.1826  0.1743  0.1661  0.1579  0.1497  0.1415
## SD  0.1924   0.182  0.1740  0.1687  0.1664  0.1672  0.1710  0.1777  0.1869
##    port 20
## ER  0.1332
## SD  0.1983
```

```
gmin.port.all
```

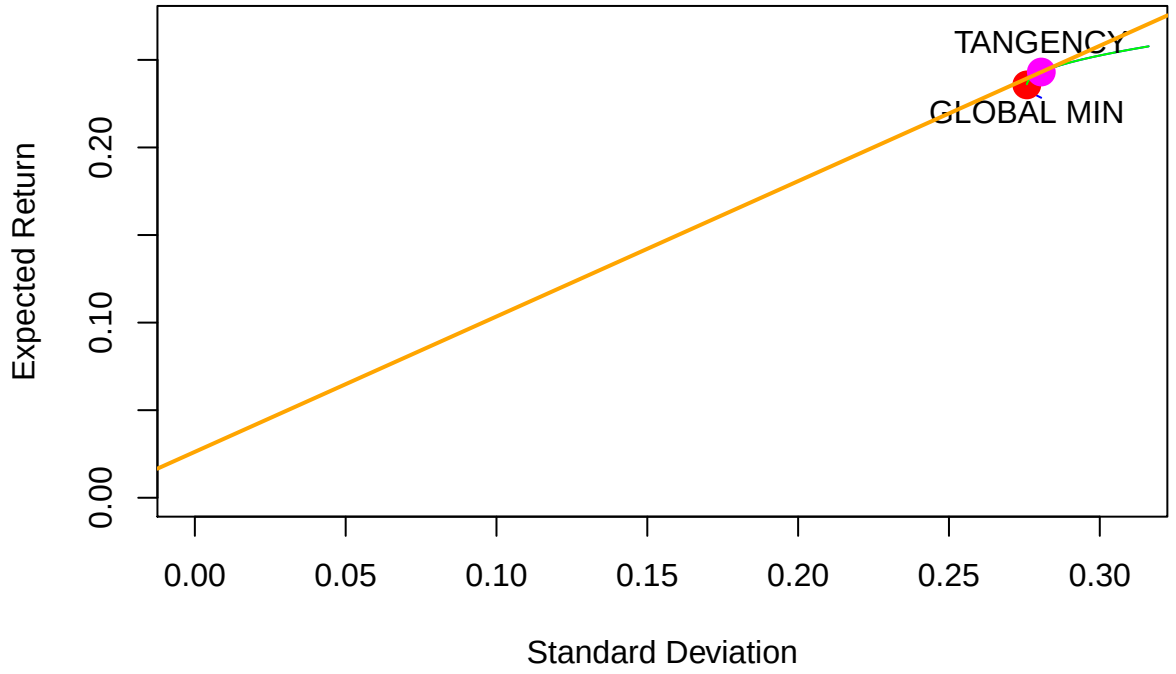
```
## Call:
## globalMin.portfolio(er = c(MSFT_mean, INTC_mean, JNJ_mean, LUV_mean,
##     MCD_mean), cov.mat = Cov_All)
##
## Portfolio expected return:     0.1722901
## Portfolio standard deviation:  0.1663273
```

```
## Portfolio weights:
## [1] 0.1083 0.0513 0.0549 0.3267 0.4588
```

From the minimum variance portfolio and the efficient frontier, we can observe that: 1. As the number of stocks increased, volatility of the portfolio decreased with almost same amount of the return 2.The efficient frontier has shifted towards the left slightly, indicating lower volatilities

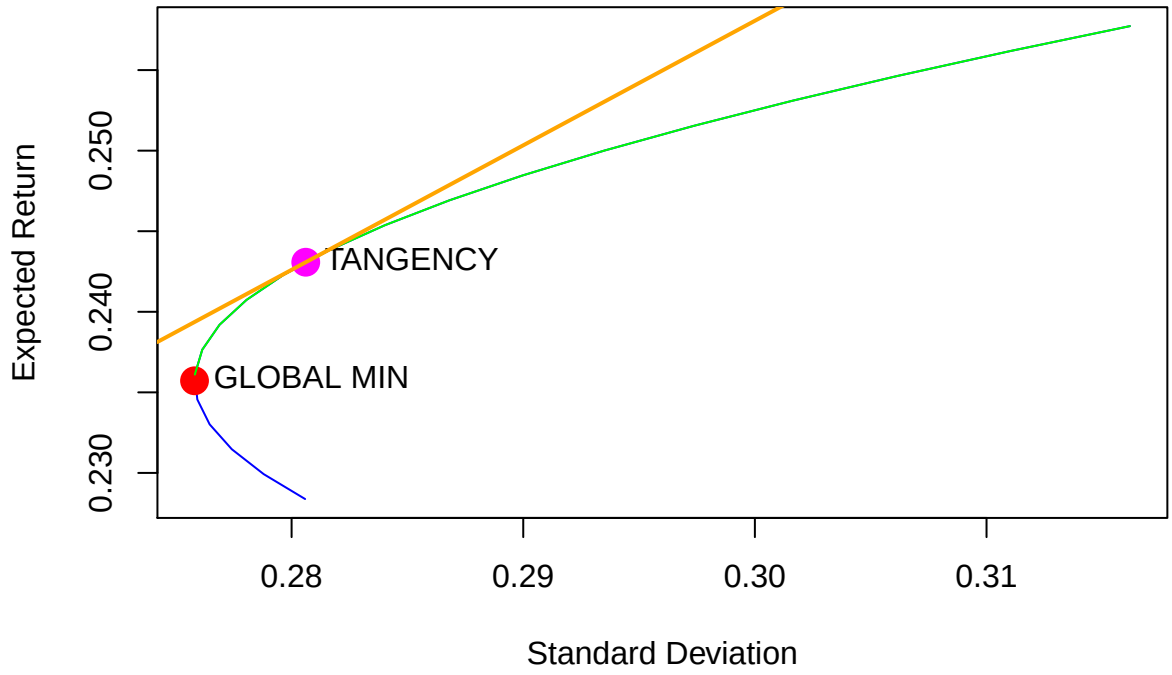
4. Add the riskless asset and construct the tangent portfolio for the Intel-Microsoft case. Next, construct the tangent portfolio for the full set of stocks. Compare the Sharpe ratios of the two tangent portfolios.

```
ef_INTC_MSFT_RF <- efficient.frontier(c(MSFT_mean,INTC_mean), Cov_MSFT_INTC)

# tangency portfolio
tan.port <- tangency.portfolio(c(MSFT_mean,INTC_mean), Cov_MSFT_INTC, RF_mean)

# compute global minimum variance portfolio
gmin.port.INTC_MSFT_RF = globalMin.portfolio(c(MSFT_mean,INTC_mean),
                                             Cov_MSFT_INTC)
plot(ef_INTC_MSFT_RF$sd, ef_INTC_MSFT_RF$er, col="blue", pch=16, type = 'l',
     xlab = "Standard Deviation", ylab = "Expected Return", xlim = c(0,0.31),
     ylim = c(0, 0.27))
points(gmin.port.INTC_MSFT_RF$sd, gmin.port.INTC_MSFT_RF$er, col="red", pch=16,
       cex=2)
lines(ef_INTC_MSFT_RF$sd[1:15], ef_INTC_MSFT_RF$er[1:15], col="green", pch=16,
      type = 'l')
points(tan.port$sd, tan.port$er, col="magenta", pch=16, cex=2)
text(gmin.port.INTC_MSFT_RF$sd, gmin.port.INTC_MSFT_RF$er, labels="GLOBAL MIN",
     pos=1)
text(tan.port$sd, tan.port$er, labels="TANGENCY", pos=3)
sr.tan = (tan.port$er - RF_mean)/tan.port$sd
abline(a=RF_mean, b=sr.tan, col="orange", lwd=2)
```



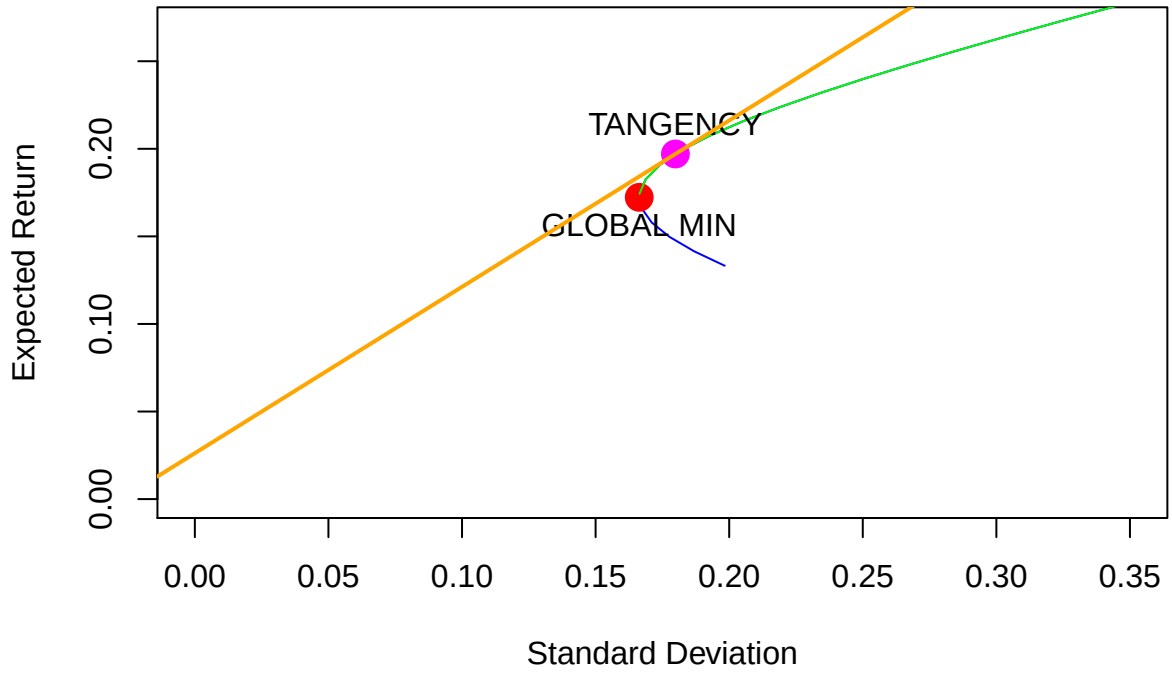

```
plot(ef_INTC_MSFT_RF$sd, ef_INTC_MSFT_RF$er, col="blue", pch=16, type = 'l',
     xlab = "Standard Deviation", ylab = "Expected Return")
```

```
points(gmin.port.INTC_MSFT_RF$sd, gmin.port.INTC_MSFT_RF$er, col="red",
       pch=16, cex=2)
lines(ef_INTC_MSFT_RF$sd[1:15], ef_INTC_MSFT_RF$er[1:15], col="green", pch=16,
      type = 'l')
points(tan.port$sd, tan.port$er, col="magenta", pch=16, cex=2)
text(gmin.port.INTC_MSFT_RF$sd, gmin.port.INTC_MSFT_RF$er,
     labels="GLOBAL MIN", pos=4)
text(tan.port$sd, tan.port$er, labels="TANGENCY", pos=4)
sr.tan = (tan.port$er - RF_mean)/tan.port$sd
abline(a=RF_mean, b=sr.tan, col="orange", lwd=2)
```

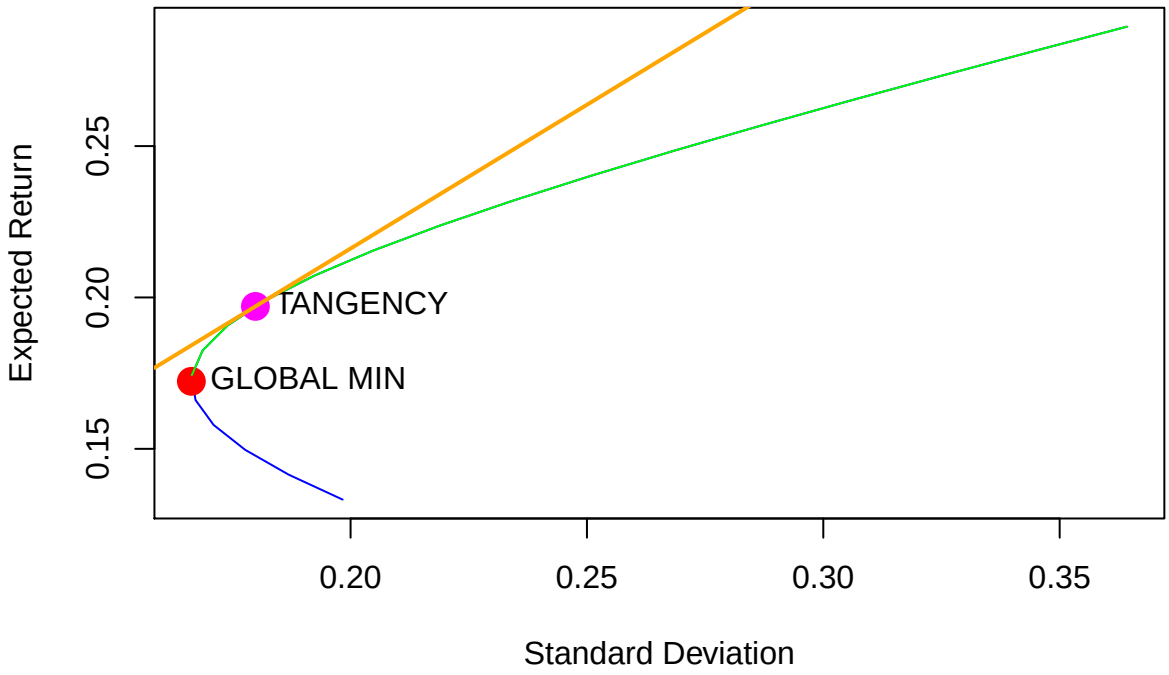



```
#All stocks with RF

ef_All_RF <- efficient.frontier(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean,
                                  MCD_mean), Cov_All)

# tangency portfolio
tan.port.all <- tangency.portfolio(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean,
                                     MCD_mean), Cov_All, RF_mean)

# compute global minimum variance portfolio
gmin.port.All_RF = globalMin.portfolio(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean,
                                         MCD_mean), Cov_All)
plot(ef_All_RF$sd, ef_All_RF$er, col="blue", pch=16, type = 'l',
     xlab = "Standard Deviation", ylab = "Expected Return", xlim = c(0,0.35),
     ylim = c(0, 0.27))
points(gmin.port.All_RF$sd, gmin.port.All_RF$er, col="red", pch=16, cex=2)
lines(ef_All_RF$sd[1:15], ef_All_RF$er[1:15], col="green", pch=16, type = 'l')
points(tan.port.all$sd, tan.port.all$er, col="magenta", pch=16, cex=2)
text(gmin.port.All_RF$sd, gmin.port.All_RF$er, labels="GLOBAL MIN", pos=1)
text(tan.port.all$sd, tan.port.all$er, labels="TANGENCY", pos=3)
sr.tan.all = (tan.port.all$er - RF_mean)/tan.port.all$sd
```

```
abline(a=RF_mean, b=sr.tan.all, col="orange", lwd=2)
```




```
gmin.port.All_RF = globalMin.portfolio(c(MSFT_mean,INTC_mean,JNJ_mean, LUV_mean,
                                         MCD_mean), Cov_All)
plot(ef_All_RF$sd, ef_All_RF$er, col="blue", pch=16, type = 'l',
     xlab = "Standard Deviation", ylab = "Expected Return")
points(gmin.port.All_RF$sd, gmin.port.All_RF$er, col="red", pch=16, cex=2)
lines(ef_All_RF$sd[1:15], ef_All_RF$er[1:15], col="green", pch=16, type = 'l')
points(tan.port.all$sd, tan.port.all$er, col="magenta", pch=16, cex=2)
text(gmin.port.All_RF$sd, gmin.port.All_RF$er, labels="GLOBAL MIN", pos=4)
text(tan.port.all$sd, tan.port.all$er, labels="TANGENCY", pos=4)
sr.tan.all = (tan.port.all$er - RF_mean)/tan.port.all$sd
abline(a=RF_mean, b=sr.tan.all, col="orange", lwd=2)
```



Sharpe Ratio for MSFT-INTC portfolio: 0.773 Sharpe Ratio for portfolio with all stocks: 0.95

We can observe that the Sharpe Ratio improved as we kept adding more stocks (as we diversify).

5. Assume your risk aversion is A = 4. What is your optimal mix of assets (including the risk-free asset)?

```
#Compute w*

w <- (tan.port.all$er - RF_mean)/(4*tan.port.all$sd^2)
```

6. Regress excess stock returns on excess market returns to obtain estimates of the betas of the five stocks. Compute the standard errors of your estimates.

```
#Microsoft
out_MSFT <- lm((MSFT_return-RF_return) ~ MktRF_return)
summary(out_MSFT)
```

```
##
## Call:
## lm(formula = (MSFT_return - RF_return) ~ MktRF_return)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.167800 -0.017241 -0.001232  0.014247  0.222749
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0027019  0.0008434   3.203  0.00138 **
## MktRF_return 0.9765890  0.0356154  27.420  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03368 on 1600 degrees of freedom
## Multiple R-squared:  0.3197, Adjusted R-squared:  0.3193
## F-statistic: 751.9 on 1 and 1600 DF,  p-value: < 2.2e-16
```

```r
#Intel
out_INTC <- lm((INTC_return-RF_return) ~ MktRF_return)
summary(out_INTC)
```

```
##
## Call:
## lm(formula = (INTC_return - RF_return) ~ MktRF_return)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.193623 -0.020839 -0.000807  0.020175  0.229637
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.001451   0.001008   1.439     0.15
## MktRF_return 1.194556   0.042567  28.063   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04025 on 1600 degrees of freedom
## Multiple R-squared:  0.3299, Adjusted R-squared:  0.3294
## F-statistic: 787.5 on 1 and 1600 DF,  p-value: < 2.2e-16
```

```r
#Microsoft
out_JNJ <- lm((JNJ_return-RF_return) ~ MktRF_return)
summary(out_JNJ)
```

```
##
## Call:
## lm(formula = (JNJ_return - RF_return) ~ MktRF_return)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.132187 -0.013623 -0.000674  0.012657  0.183137
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0013050  0.0006293   2.074   0.0383 *
## MktRF_return 0.5633166  0.0265711  21.200   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02512 on 1600 degrees of freedom
## Multiple R-squared:  0.2193, Adjusted R-squared:  0.2188
## F-statistic: 449.5 on 1 and 1600 DF,  p-value: < 2.2e-16
```

```r
#Microsoft
out_LUV <- lm((LUV_return-RF_return) ~ MktRF_return)
summary(out_LUV)
```

```
##
## Call:
## lm(formula = (LUV_return - RF_return) ~ MktRF_return)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -0.18100 -0.02336 -0.00056  0.02217  0.21990
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.001155   0.001006   1.149    0.251
## MktRF_return 1.080950   0.042461  25.457   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04015 on 1600 degrees of freedom
## Multiple R-squared:  0.2883, Adjusted R-squared:  0.2878
## F-statistic: 648.1 on 1 and 1600 DF,  p-value: < 2.2e-16
```

```
#Microsoft
out_MCD <- lm((MCD_return-RF_return) ~ MktRF_return)
summary(out_MCD)
```

```
##
## Call:
## lm(formula = (MCD_return - RF_return) ~ MktRF_return)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.125025 -0.014897 -0.000489  0.014313  0.115913
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0013048  0.0006855   1.903   0.0572 .
## MktRF_return 0.6431096  0.0289453  22.218   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02737 on 1600 degrees of freedom
## Multiple R-squared:  0.2358, Adjusted R-squared:  0.2353
## F-statistic: 493.6 on 1 and 1600 DF,  p-value: < 2.2e-16
```

Estimate of Betas for MSFT, INTC, JNJ, LUV, MCD; in order: 0.9765890, 1.194556, 0.5633166, 1.080950, 0.6431096

7. What are the estimates of the alphas, and the standard deviation of these estimates? What are the estimates of the standard deviation of idiosyncratic risk?

Estimate of the standard deviation of idiosyncratic risk is just same as the standard error of regression.

Estimate of Alpha for MSFT: 0.0027019 Standard deviation of Alpha for MSFT: 0.0008434 Estimates of the standard deviation of idiosyncratic risk for MSFT: 0.03368

Estimate of Alpha for INTC: 0.001451 Standard deviation of Alpha for INTC: 0.001008 Estimates of the standard deviation of idiosyncratic risk for INTC: 0.04025

Estimate of Alpha for JNJ: 0.0013050 Standard deviation of Alpha for JNJ: 0.0006293 Estimates of the standard deviation of idiosyncratic risk for JNJ: 0.02512

Estimate of Alpha for LUV: 0.001155 Standard deviation of Alpha for LUV: 0.001006 Estimates of the standard deviation of idiosyncratic risk for LUV: 0.04015

Estimate of Alpha for MCD: 0.0013048 Standard deviation of Alpha for MCD: 0.0006855 Estimates of the standard deviation of idiosyncratic risk for MCD: 0.02737

8. Compute the sample average excess return and compare the value with the return predicted by the CAPM. Based on the data, how well does the CAPM predict the level returns? How well does the CAPM predict relative performance?

```
MSFT_mean - RF_mean
```

```
## [1] 0.2241821
```

```
0.9765890*MktRF_mean
```

```
## [1] 0.08368289
```

```
INTC_mean - RF_mean
```

```
## [1] 0.1778056
```

```
1.194556*MktRF_mean
```

```
## [1] 0.1023603
```

```
JNJ_mean - RF_mean
```

```
## [1] 0.1161297
```

```
0.5633166*MktRF_mean
```

```
## [1] 0.04827001
```

```
LUV_mean - RF_mean
```

```
## [1] 0.1527046
```

```
1.080950*MktRF_mean
```

```
## [1] 0.09262547
```

```
MCD_mean - RF_mean
```

```
## [1] 0.122957
```

```
0.6431096*MktRF_mean
```

```
## [1] 0.05510739
```

Sample Average Excess Return of MSFT, INTC, JNJ, LUV, MCD; in order: 0.2241821, 0.1778056, 0.1161297, 0.1527046, 0.122957 Return Predicted by CAPM for MSFT, INTC, JNJ, LUV, MCD; in order: 0.08368289, 0.1023603, 0.04827001, 0.09262547, 0.05510739

Based on the data, we can see that CAPM does not properly predict the level returns and relative performance.