HW 4
MFE 407: Empirical Methods in Finance
Professor Lochstoer
Group 6
Students: Xiahao Wang, Juan Manuel Ferreyra Maspero, Xinyue Zhu, Yichu Li, Mu Lin

Problem 1

1. To find a parsimonious model for the conditional volatility of the daily log return.

Step 1:

Find the daily log return using the following code:

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

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

```
library(rugarch)
```

```
## Warning: package 'rugarch' was built under R version 3.5.2
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.5.2
```

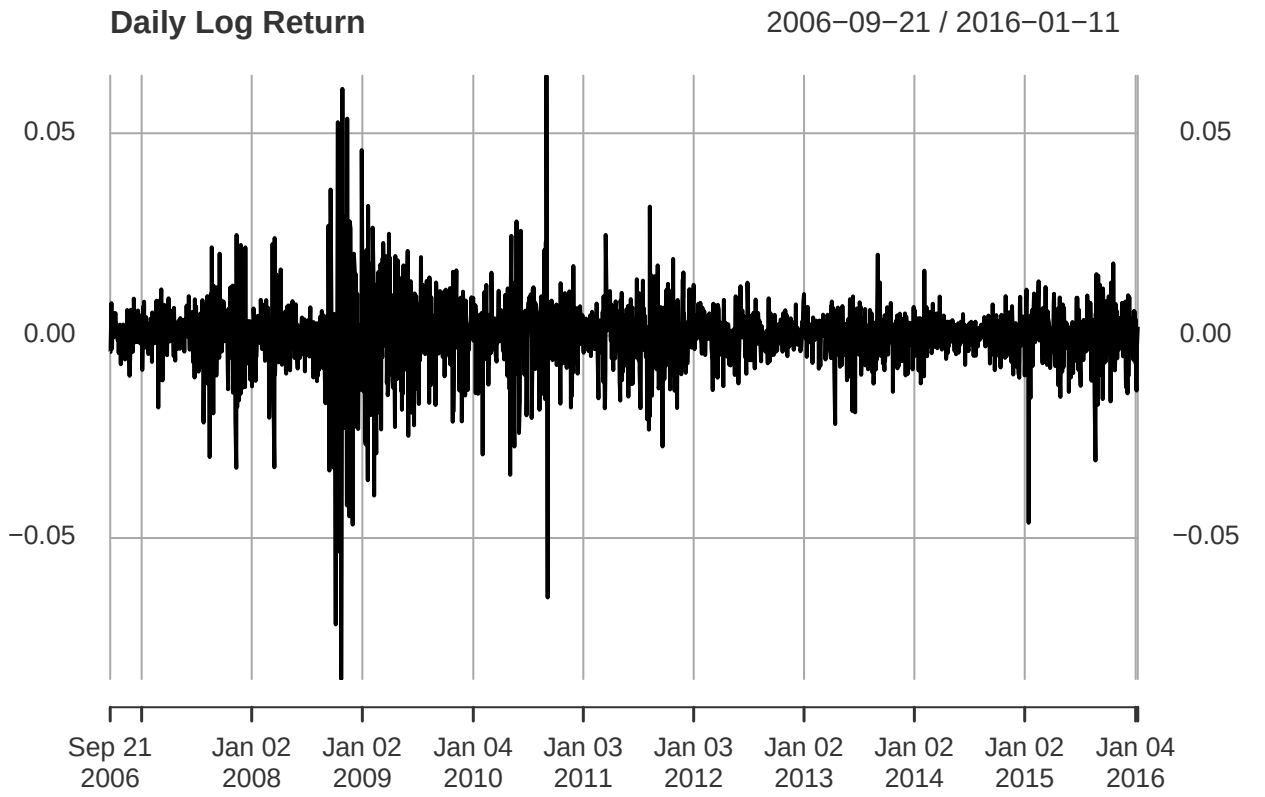
```
library(lmtest)
library(TSA)
```

```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

```
## The following object is masked from 'package:utils':
##
##     tar
```

```
# Read in data
rawdata <- read.csv("Currency_fund_prices.csv", header=TRUE, sep = ",")

# Get log Return
fundPrices <- rawdata[,c(1,7)]
logRet <- diff(log(fundPrices$Adj.Close))
logRet_vol <- xts(x=logRet, dmy(fundPrices$Date[-1]))

plot(logRet_vol,type ="l",main="Daily Log Return")
```

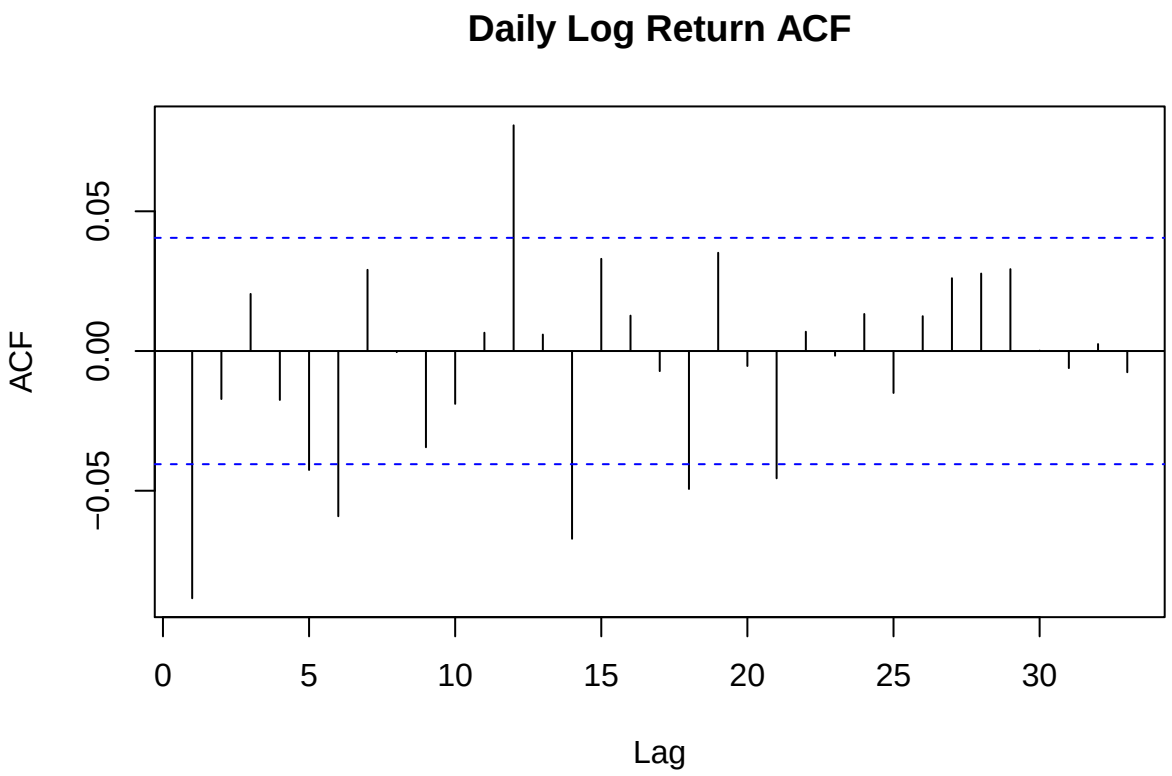



```
logRet_vol <- xts(x=logRet_vol, dmy(fundPrices$Date[-1]))
```

Step 2:

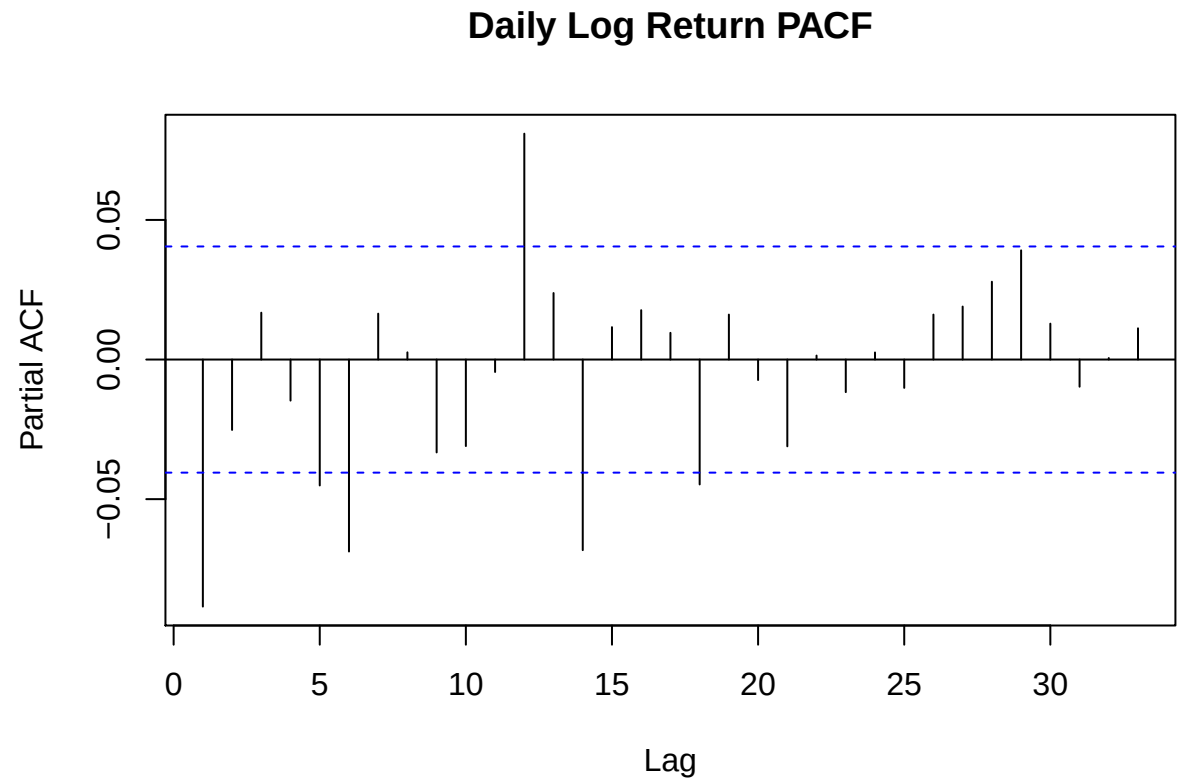
Determine the ARMA model for the log return process ACF plot of the daily log return:

```
acf(logRet_vol,main="Daily Log Return ACF")
```



PACF plot of the daily log return:

```
pacf(logRet_vol,main="Daily Log Return PACF")
```



Based on PACF and ACF, it is hard to determine the AR MA terms.

In this case, we use auto.arima function to determine the best model based on AIC and BIC

```
auto_model <- auto.arima(logRet_vol)
summary(auto_model)
```

```
## Series: logRet_vol
## ARIMA(3,0,3) with zero mean
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2      ma3
##       0.9508  -0.9742  0.7063  -1.0480  1.0456  -0.7501
## s.e.  0.0860   0.0514  0.0706   0.0825  0.0557   0.0620
##
## sigma^2 estimated as 7.519e-05:  log likelihood=7799.02
## AIC=-15584.04   AICc=-15583.99   BIC=-15543.72
##
## Training set error measures:
##                       ME        RMSE         MAE MPE MAPE      MASE
## Training set -2.7373e-05 0.008660061 0.005591712 NaN  Inf 0.6777815
##                      ACF1
## Training set 0.003148712
```

```
coeftest(auto_model)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1  0.950773   0.086033  11.051 < 2.2e-16 ***
## ar2 -0.974158   0.051412 -18.948 < 2.2e-16 ***
## ar3  0.706348   0.070630  10.001 < 2.2e-16 ***
## ma1 -1.047984   0.082451 -12.710 < 2.2e-16 ***
## ma2  1.045623   0.055694  18.774 < 2.2e-16 ***
## ma3 -0.750139   0.062013 -12.097 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARMA(3,0,3) is suggested and all the coefficients are statistically significant.

Step 3:

Determine the best garch model to use: We use ugarchfit to check the model. We will be checking whether the coefficients are statistically significant.
We also need to check AIC/BIC, Weighted Ljung-Box Test on Standardized Residuals.

After trying out a few models, we determine the model eGarch(1,1) with t distribution to be the most appropriate model so far given the statistics.

```
garch_model <- ugarchspec(variance.model = list(model = "eGARCH",garchOrder = c(1,1)),
                          mean.model = list(armaOrder = c(3,0,3)),
                          distribution.model = "std")

(garchfit <- ugarchfit(data = logRet_vol, spec = garch_model))
```

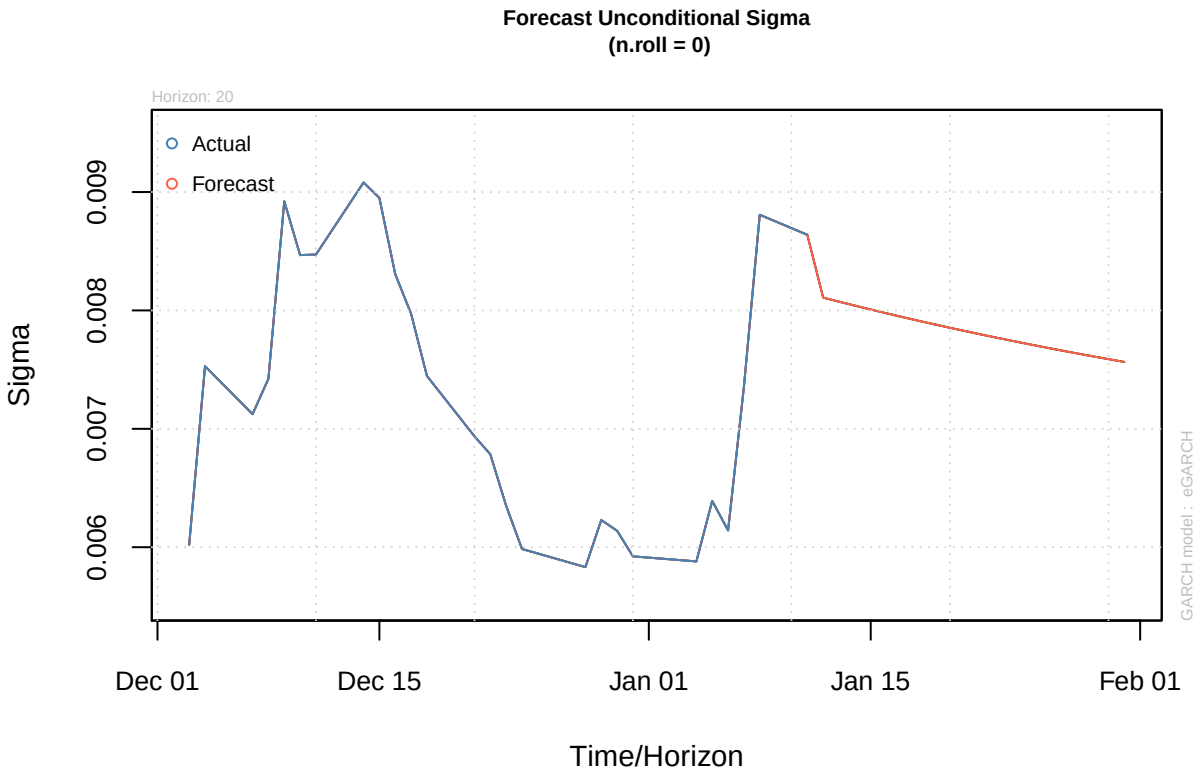
```
##
## *---------------------------------*
```

```
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## ---------------------------------
## GARCH Model  : eGARCH(1,1)
## Mean Model   : ARFIMA(3,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000296    0.000091   3.24653 0.001168
## ar1    -0.048432    0.017476  -2.77144 0.005581
## ar2     0.006107    0.012474   0.48959 0.624422
## ar3     0.016636    0.008679   1.91667 0.055279
## omega  -0.169313    0.010086 -16.78694 0.000000
## alpha1 -0.063067    0.015382  -4.10013 0.000041
## beta1   0.983296    0.001128 872.08444 0.000000
## gamma1  0.200514    0.006909  29.02189 0.000000
## shape   5.319338    0.589310   9.02639 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000296    0.000081   3.64680 0.000266
## ar1    -0.048432    0.014268  -3.39444 0.000688
## ar2     0.006107    0.007351   0.83075 0.406114
## ar3     0.016636    0.003994   4.16477 0.000031
## omega  -0.169313    0.012557 -13.48402 0.000000
## alpha1 -0.063067    0.016290  -3.87158 0.000108
## beta1   0.983296    0.001172 838.91430 0.000000
## gamma1  0.200514    0.009687  20.70019 0.000000
## shape   5.319338    0.656910   8.09751 0.000000
##
## LogLikelihood : 8480.638
##
## Information Criteria
## ------------------------------------
##
## Akaike       -7.2345
## Bayes        -7.2124
## Shibata      -7.2346
## Hannan-Quinn -7.2265
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.06344  0.8011
## Lag[2*(p+q)+(p+q)-1][8]   1.28871  1.0000
## Lag[4*(p+q)+(p+q)-1][14]  2.44336  0.9995
## d.o.f=3
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
```

```
## -------------------------------------
##                         statistic p-value
## Lag[1]                      5.236 0.02212
## Lag[2*(p+q)+(p+q)-1][5]     6.971 0.05270
## Lag[4*(p+q)+(p+q)-1][9]     9.558 0.06225
## d.o.f=2
##
## Weighted ARCH LM Tests
## -------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.02574 0.500 2.000  0.8725
## ARCH Lag[5]   4.00918 1.440 1.667  0.1731
## ARCH Lag[7]   4.97354 2.315 1.543  0.2272
##
## Nyblom stability test
## -------------------------------------
## Joint Statistic:  3.037
## Individual Statistics:
## mu     1.00435
## ar1    0.36907
## ar2    0.03315
## ar3    0.17307
## omega  0.48541
## alpha1 1.01294
## beta1  0.46447
## gamma1 0.21760
## shape  0.06416
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.1 2.32 2.82
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## -------------------------------------
##                    t-value    prob sig
## Sign Bias          2.33921 0.01941  **
## Negative Sign Bias 0.09733 0.92248
## Positive Sign Bias 1.30375 0.19245
## Joint Effect       6.56492 0.08714   *
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     42.61    0.0014645
## 2    30     53.49    0.0037004
## 3    40     68.69    0.0023186
## 4    50     85.46    0.0009755
##
##
## Elapsed time : 0.6625609
```

2. Using the model above, we have forecast the 20-trading-day log return volatility

```
garchForecast <- ugarchforecast(garchfit,n.ahead = 20)
plot(garchForecast,which=3)
```




```
cat("20-trading-day log return volatility
on Jan 11, 2016 (end of day) is: ", sqrt(sum(sigma(garchForecast)^2)))
```

```
## 20-trading-day log return volatility
## on Jan 11, 2016 (end of day) is:  0.03497928
```

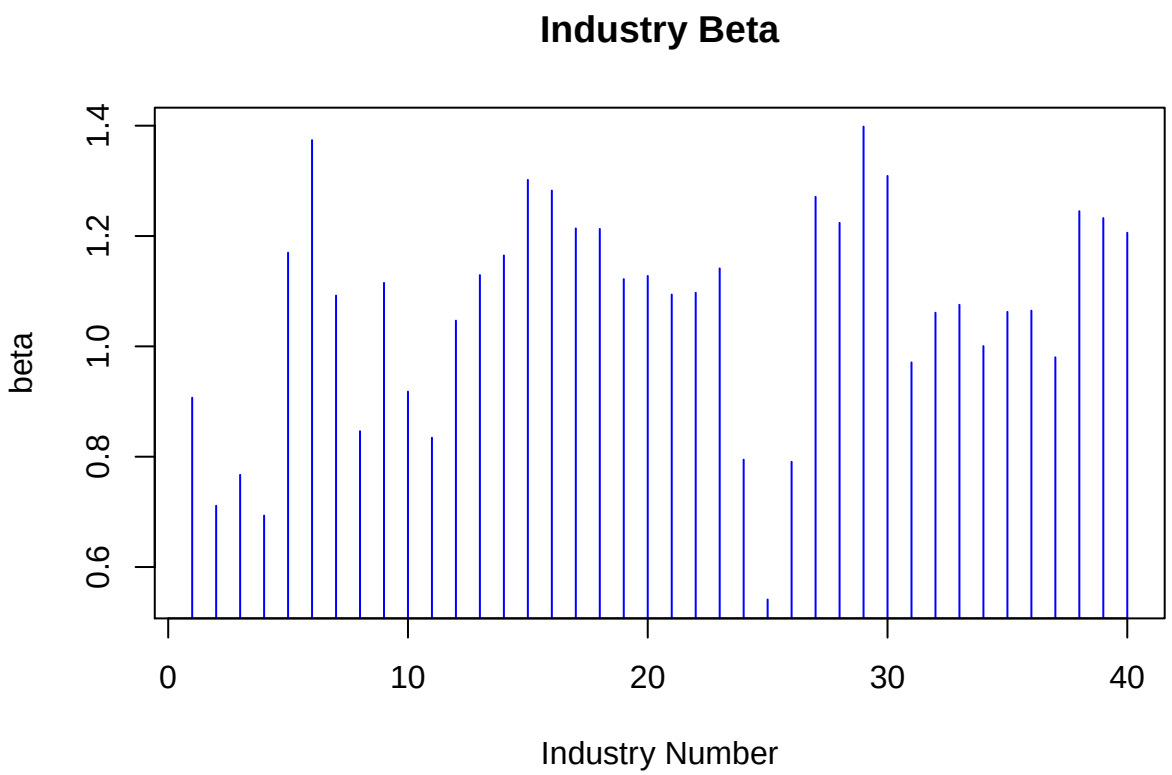
Problem 2

1.

(a)

```
industry <- read.csv("48_Industry_Portfolios.csv")
three_fac <- read.csv("F-F_Research_Data_Factors.csv")


ind_ret<-industry[which(industry$X==196001):which(industry$X==201512),] # extract relevant year
ind_ret<-ind_ret[,-c(4,12,16,21,27,28,34,39)] # remove -99.99
three_fac<-as.data.frame(three_fac[which(three_fac$X==196001):which(three_fac$X==201512),]) # extract r
# to merge data
data2 <- merge(ind_ret,three_fac,by="X")

beta<-r_sqr<-alpha<-c(0)
num_ind<-length(ind_ret[1,])-1

for(i in 1:num_ind){
  ex_ret=data2[,i+1]-as.numeric(as.character(data2$RF))
```

```
  out=lm(ex_ret~as.numeric(as.character(data2$Mkt.RF)))
  beta[i]=out$coef[2]
  alpha[i]=out$coef[1]
  r_sqr[i]=anova(out)[1,2]/(anova(out)[1,2]+anova(out)[2,2]) #SSR/SST
}

# 2 (a) Plot the industry betas
plot(seq(1,num_ind,1),col="blue",y=beta,pch=14,type="h",main="Industry Beta",xlab="Industry Number")
abline(h=2*sd(beta),col="red")
abline(h=-2*sd(beta),col="red")
```



(b)

```
# 2 (b) What is the range of estimated betas? What are the min, max, and mean regression R^2 across ind
cat("The range of the estimated betas is from the min", min(beta),"\n to the max",max(beta),"with mean
```

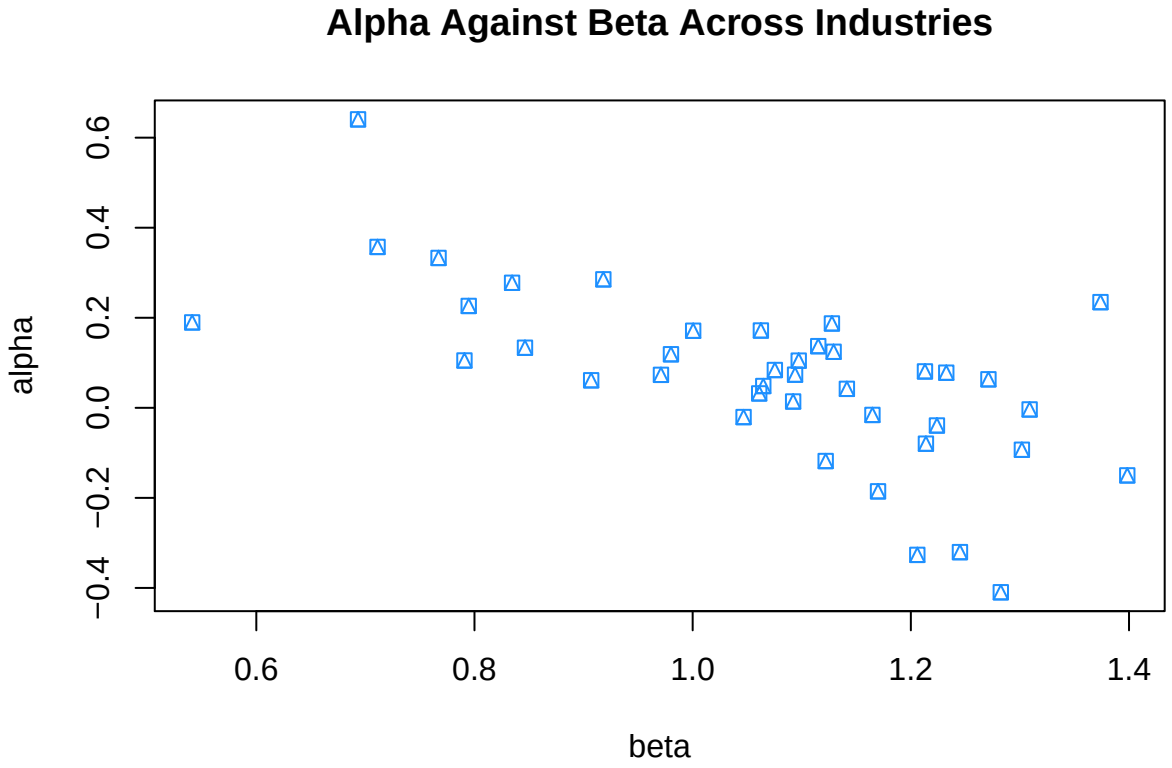
```
## The range of the estimated betas is from the min 0.5412037
##  to the max 1.398412 with mean of  1.064737
```

```
cat("The regressed R^2 accross industries is from the min", min(r_sqr),"\n to the max",max(r_sqr),"with
```

```
## The regressed R^2 accross industries is from the min 0.2505185
##  to the max 0.8007488 with mean of  0.575773
```

(c)

```
# 2 (c) Plot estimated alphas (intercept terms) against estimated betas
plot(y=alpha,x=beta,type="p",pch=14,col="dodgerblue",
     main="Alpha Against Beta Across Industries")
```

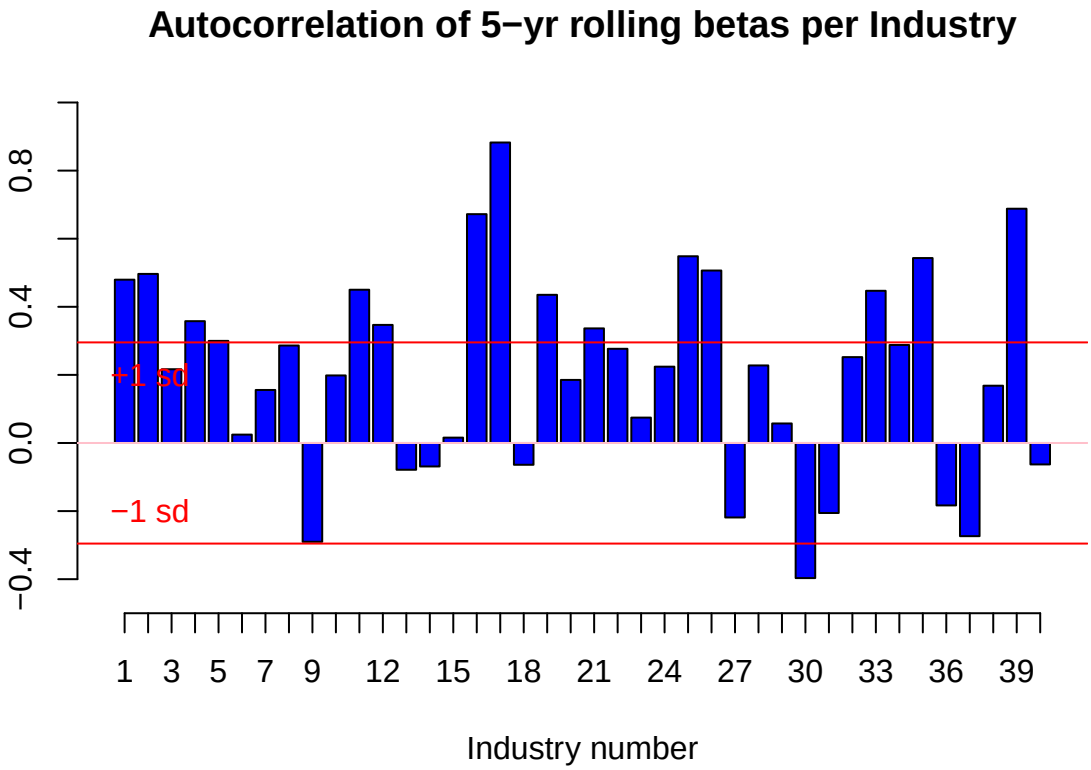


It appears the higher the beta, the smaller the alpha. It is noted that the alpha does not appear to converge to 0, which is assumed by the CAPM model.

2.

```
library(DataAnalytics)
# 3. Run rolling regressions of 5 years of data.
beta_matrix<-matrix(0,num_ind,11)
for(i in 1:num_ind){
  for(k in 1:11){
  ex_ret=data2[(60*(k-1)+1):(60*k-1),(i+1)] -
    as.numeric(as.character(data2$RF[(60*(k-1)+1):(60*k-1)]))
  mkt=as.numeric(as.character(data2$Mkt.RF[(60*(k-1)+1):(60*k-1)]))
  out=lm(ex_ret~mkt)
  beta_matrix[i,k]=out$coef[2]
  }
}

# Autocorrelation of betas per adjacent 5-yr rolling windows per industry
# AR(1)
beta_corr<-c(0)
for(i in 1:(num_ind)){
  beta_corr[i]=lm(beta_matrix[i,]~back(beta_matrix[i,]))$coef[2]
}

myplot <- barplot(beta_corr, col="blue", ,xlab="Industry number",
                  main="Autocorrelation of 5-yr rolling betas per Industry",
                  ylim=c(-0.5,1))
axis(1, at = myplot, labels = 1:num_ind)
abline(h=0,col="pink")
abline(h=sd(beta_corr),col="red")
```

```
abline(h=-sd(beta_corr),col="red")
text(c(10,0.2),"+1 sd",col="red")
text(c(10,-0.2),"-1 sd",col="red")
```



Per our observation, betas for some industries (eg, Industry 6, FUN) tend to lie within +/-1 sd around 0. For some other industries (eg, Industry 17, Mach), however, their betas tend to be highly correlated to prior period and appear to be stable over time. One among all possible reasons to explain such divergence in stability results from the fact that some industries do not appear to have extensive exposures to market movements. Therefore, market fluctuation only impacts the stock performance of such industries to some extend, which yields less comovements and unstable betas.