UBS TakeHomeAssigment

Rongpeng Liu

October 31, 2019

Question 1: Outlier detection and statistics

(a) Load the data of the csv file into a data format of your choice.

```r
library(data.table)

# Load data, change column names, set as data.table and change data type
DT <- as.data.table(read.csv("data.csv", header = F))
DT <- t(DT)
colnames(DT) <- c("date", "Equity1", "Equity2", "VIX", "libor_1M")
DT <- as.data.table(DT[-1,])
DT[] <- lapply(DT, function(x) as.numeric(x))
DT$day=as.character("01")
DT$date <- as.Date(with(DT, paste(date, day)), "%Y%m%d")
DT <- DT[,-6]

head(DT, 5)
```

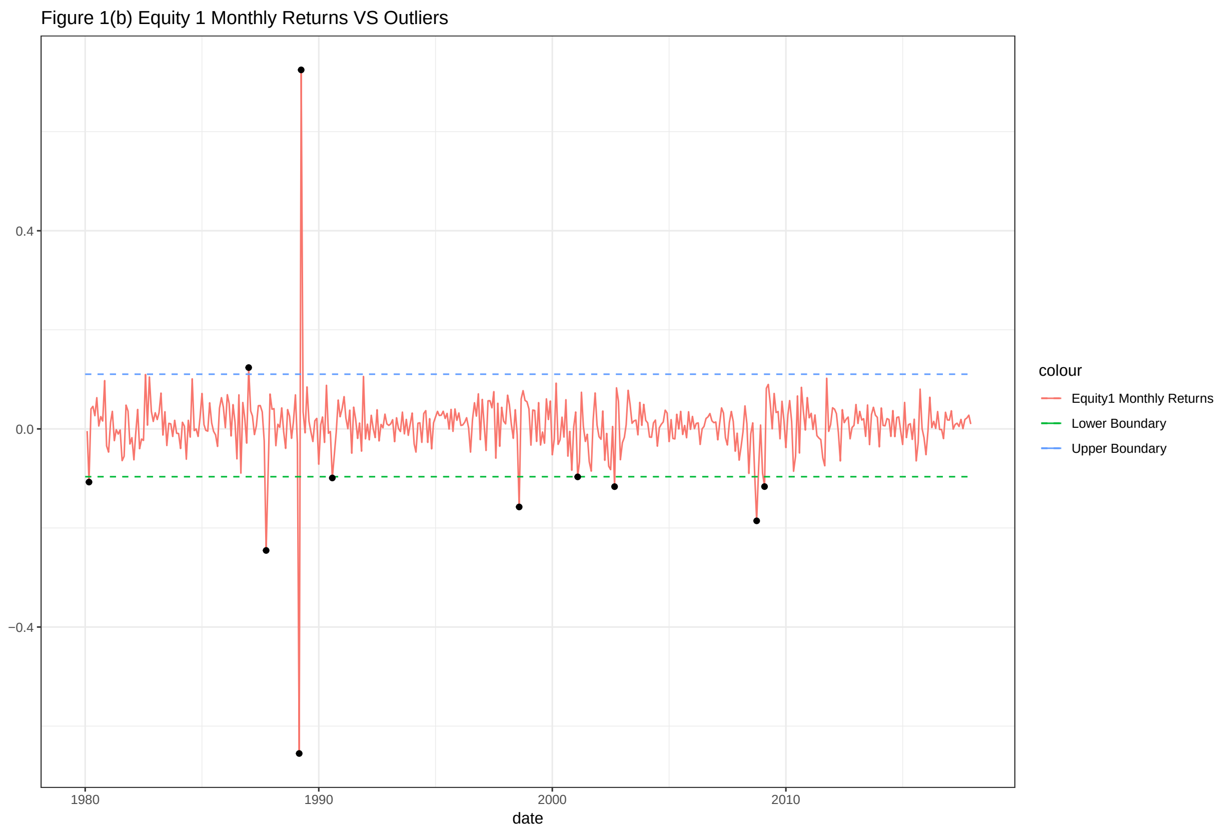
```
##          date Equity1 Equity2 VIX libor_1M
## 1: 1980-01-01  114.16      NA  NA       NA
## 2: 1980-02-01  113.66      NA  NA       NA
## 3: 1980-03-01  102.09     131  NA       NA
## 4: 1980-04-01  106.29      NA  NA       NA
## 5: 1980-05-01  111.24      NA  NA       NA
```

After loading data, the first 5 rows are shown above.

(b) Detect potential outliers for each time series based on a simple methodology.

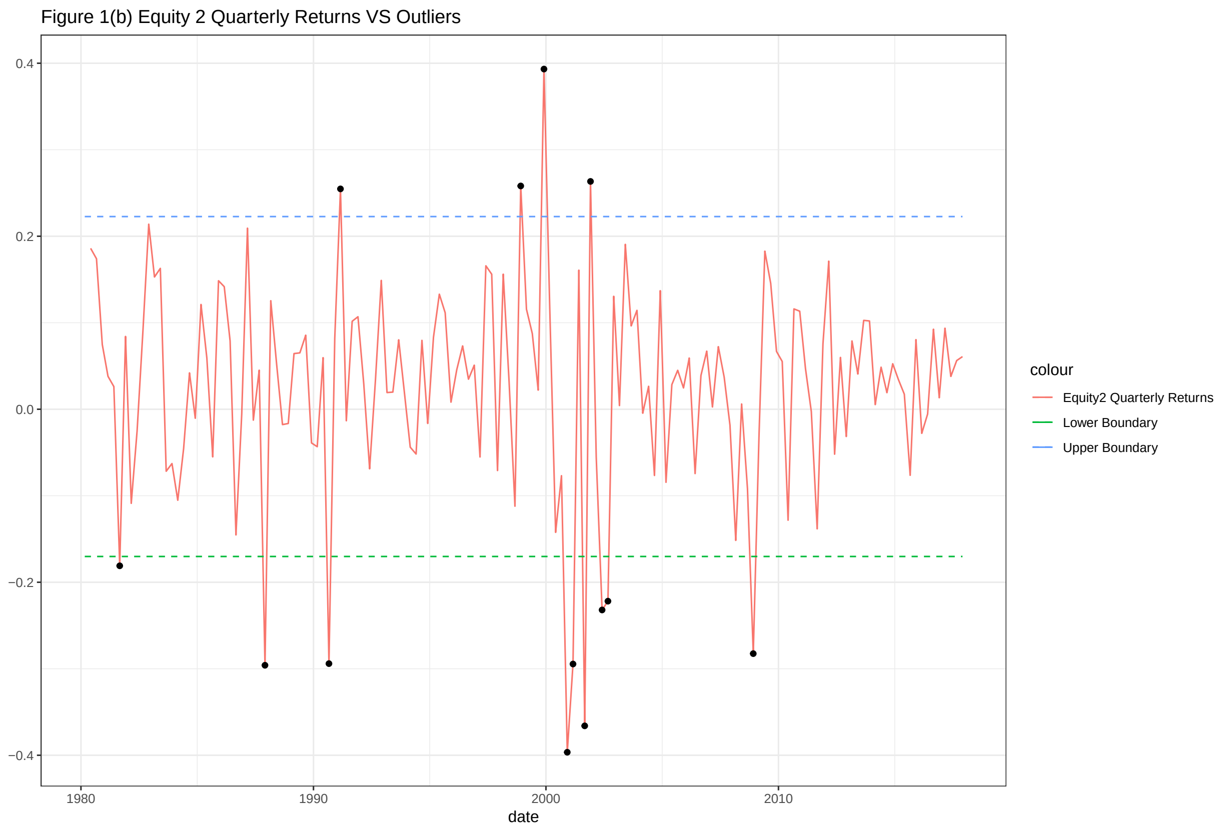
```r
library(ggplot2)

# find monthly returns for Equity 1
DT$Equity1_ret <- log(DT$Equity1/shift(DT$Equity1))

# find upper/lower boundaries for Equity1_ret
Equity1_ret_mean <- mean(DT$Equity1_ret, na.rm = T)
Equity1_ret_sd <- sd(DT$Equity1_ret, na.rm = T)
Equity1_ret_upper <- Equity1_ret_mean + 1.645 * Equity1_ret_sd
Equity1_ret_lower <- Equity1_ret_mean - 1.645 * Equity1_ret_sd

# find outliers for Equity1_ret
DT$E1_outlier <- ifelse((DT$Equity1_ret>Equity1_ret_upper) | (DT$Equity1_ret<Equity1_ret_lower),
                        DT$Equity1_ret,NA)

E1_outlier_n <- sum(!is.na(DT$E1_outlier))
```

```r
# plot Equity1_ret and outliers
ggplot(DT,aes(date)) +
  geom_line(aes(y = Equity1_ret, col = "Equity1 Monthly Returns")) +
  geom_line(aes(y = Equity1_ret_upper, col = "Upper Boundary"), linetype = "dashed") +
  geom_line(aes(y = Equity1_ret_lower, col = "Lower Boundary"), linetype = "dashed") +
  geom_point(aes(y = E1_outlier)) +
  ggtitle("Figure 1(b) Equity 1 Monthly Returns VS Outliers") +
  theme_bw() + xlab("date") + ylab("")
```



For Equity 1 Index, we calculate its log return each month and find it is a mean-reversion process. Therefore, we can assume Equity 1 returns are normally distributed and define the outliers as those points out of 90% probability interval around its mean. Particularly, the upper boundary is $\mu + 1.645 * \sigma$ and the lower boundary is $\mu - 1.645 * \sigma$. Finally, we find 11 outliers for Equity 1 index and mark them as black points.

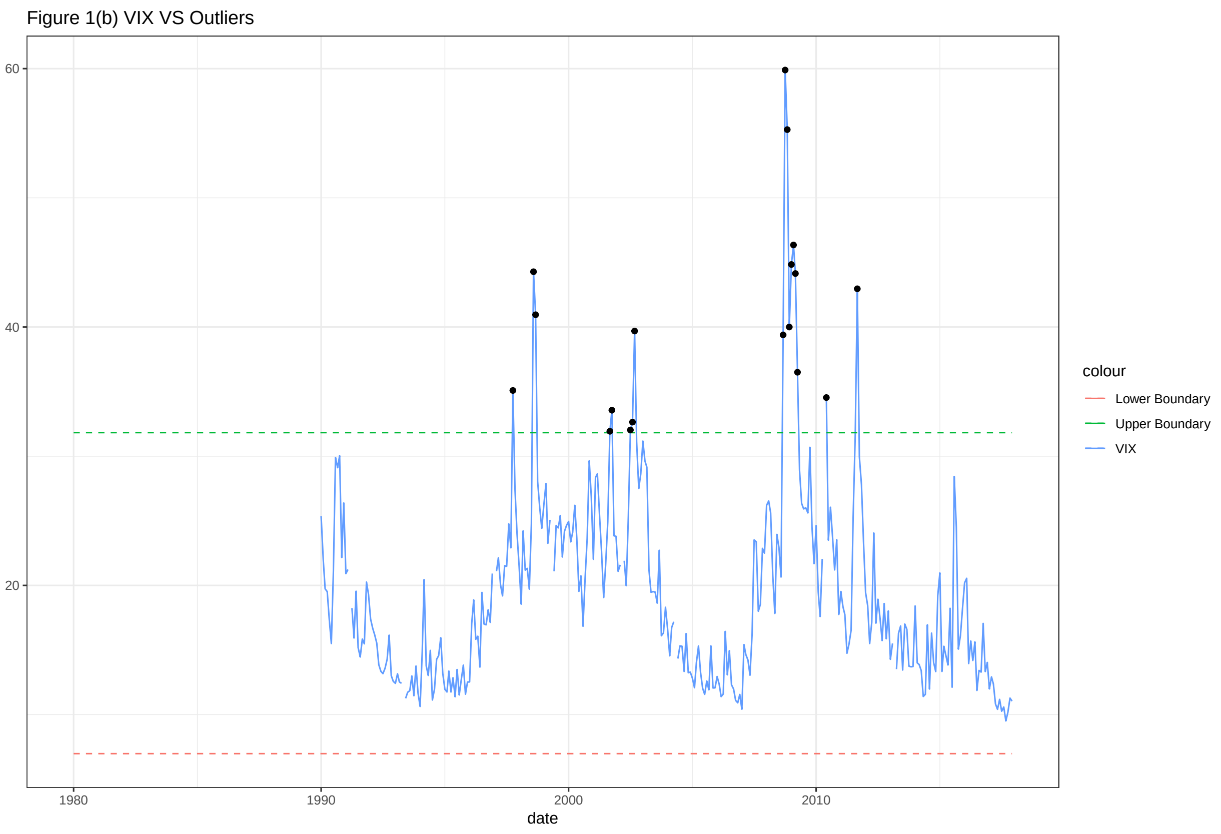
```r
# find quarterly returns for Equity 2
DT_quarter <- DT[(month(date)%%3)==0,]
DT_quarter$Equity2_ret <- log(DT_quarter$Equity2/shift(DT_quarter$Equity2))

# find upper/lower boundaries for Equity2_ret
Equity2_ret_mean <- mean(DT_quarter$Equity2_ret, na.rm = T)
Equity2_ret_sd <- sd(DT_quarter$Equity2_ret, na.rm = T)
Equity2_ret_upper <- Equity2_ret_mean + 1.645 * Equity2_ret_sd
Equity2_ret_lower <- Equity2_ret_mean - 1.645 * Equity2_ret_sd

# find outliers for Equity2_ret
DT_quarter$E2_outlier <- ifelse((DT_quarter$Equity2_ret>Equity2_ret_upper) |
                                  (DT_quarter$Equity2_ret<Equity2_ret_lower),
                                DT_quarter$Equity2_ret,NA)

E2_outlier_n <- sum(!is.na(DT_quarter$E2_outlier))

# plot Equity2_ret and outliers
ggplot(DT_quarter,aes(date)) +
  geom_line(aes(y = Equity2_ret, col = "Equity2 Quarterly Returns")) +
  geom_line(aes(y = Equity2_ret_upper, col = "Upper Boundary"), linetype = "dashed") +
  geom_line(aes(y = Equity2_ret_lower, col = "Lower Boundary"), linetype = "dashed") +
  geom_point(aes(y = E2_outlier)) +
  ggtitle("Figure 1(b) Equity 2 Quarterly Returns VS Outliers") +
  theme_bw() + xlab("date") + ylab("")
```



For Equity 2 Index, we use the same method as Equity 1 Index. The only difference is that we calculate the quarterly returns instead of monthly returns. Finally, we find 13 outliers for Equity 2 index.

```r
# find upper/lower boundaries for VIX
VIX_mean <- mean(DT$VIX, na.rm = T)
VIX_sd <- sd(DT$VIX, na.rm = T)
VIX_upper <- VIX_mean + 1.645 * VIX_sd
VIX_lower <- VIX_mean - 1.645 * VIX_sd

# find outliers for VIX
DT$VIX_outlier <- ifelse((DT$VIX>VIX_upper) | (DT$VIX<VIX_lower), DT$VIX, NA)

VIX_outlier_n <- sum(!is.na(DT$VIX_outlier))
```

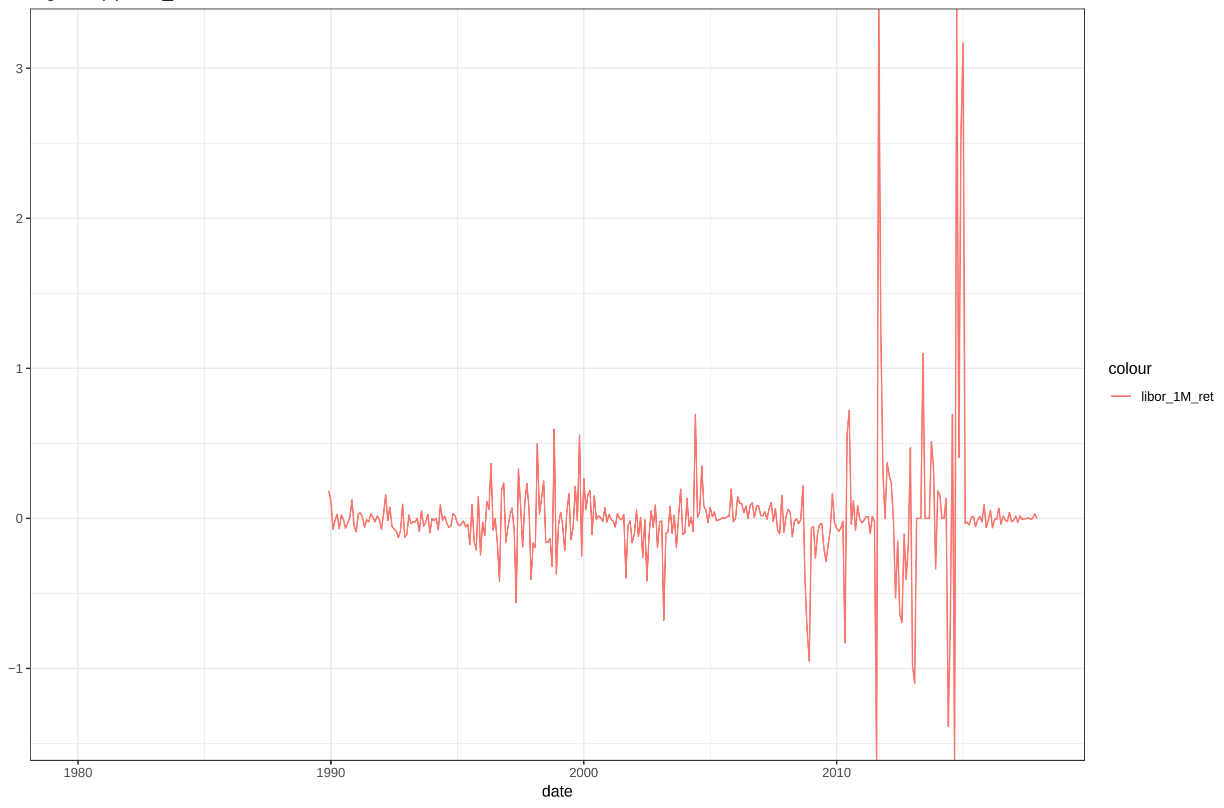
```r
# plot VIX and outliers
ggplot(DT,aes(date)) +
  geom_line(aes(y = VIX, col = "VIX")) +
  geom_line(aes(y = VIX_upper, col = "Upper Boundary"), linetype = "dashed") +
  geom_line(aes(y = VIX_lower, col = "Lower Boundary"), linetype = "dashed") +
  geom_point(aes(y = VIX_outlier)) +
  ggtitle("Figure 1(b) VIX VS Outliers") +
  theme_bw() + xlab("date") + ylab("")
```



For VIX, althouth not normally distributed, for simplicity, we still use the 90% interval method to define outliers. The only difference from previous methods is we do not need to calculate log differences. Finally, we find 18 outliers for VIX.

```r
# find returns for absolute value of Libor_1M
DT3 <- DT
DT3$libor_1M<-abs(DT3$libor_1M)
DT3$libor_1M_ret<-log(DT3$libor_1M/shift(DT3$libor_1M))

# plot returns for Libor_1M
ggplot(DT3,aes(date)) +
  geom_line(aes(y = DT3$libor_1M_ret, col = "libor_1M_ret")) +
  ggtitle("Figure 1(b) libor_1M Returns") +
  theme_bw() + xlab("date") + ylab("")
```

Figure 1(b) libor_1M Returns

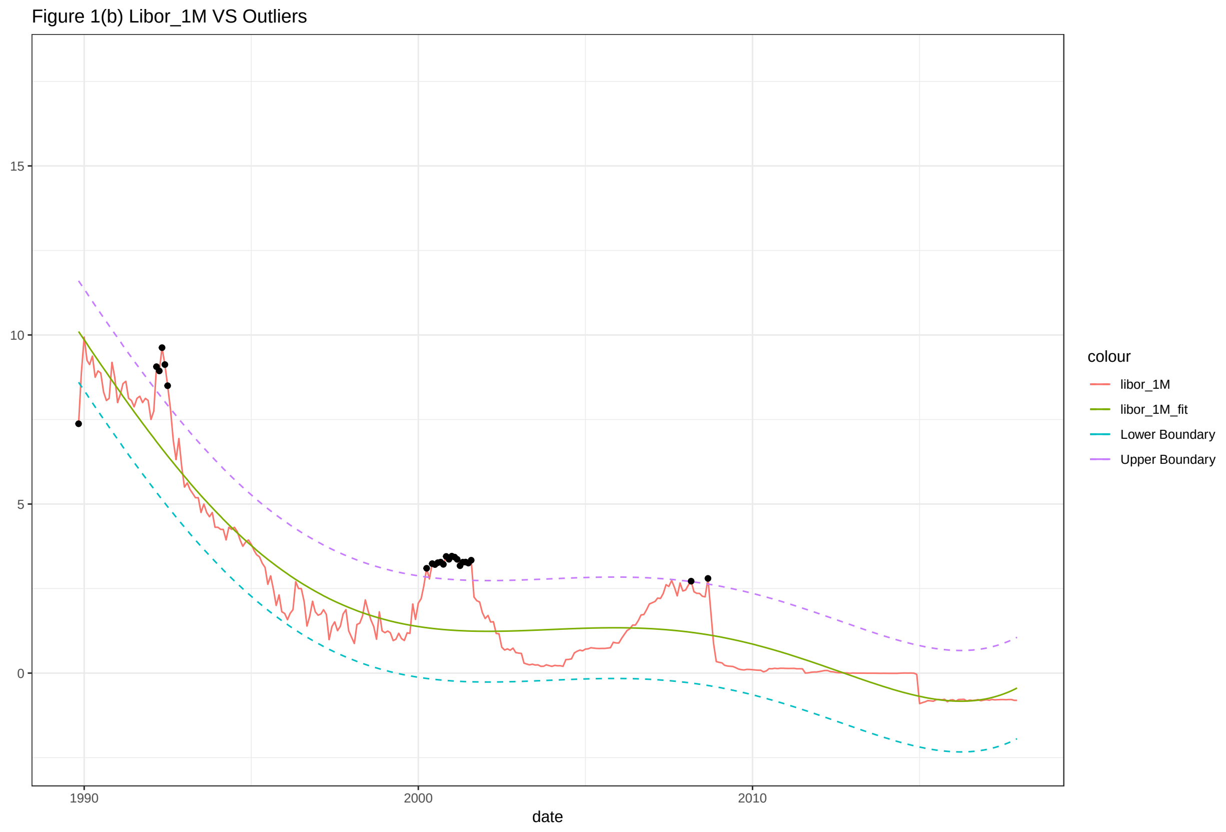


For Libor_1M, we first plot the returns by using log difference. The graph shows it has clustering property and its volatility changes over time, trivial before year 1995, but infinity for some years after 2010. So we can not use the outlier detection method of Equity 1 & 2.

```
# add year index for DT
DT$index<-1:456

# use polynomial regression to fit Libor_1M
fit1b <- lm(DT$libor_1M ~ poly(DT$index, 5, raw=TRUE))
summary(fit1b)
```

```
##
## Call:
## lm(formula = DT$libor_1M ~ poly(DT$index, 5, raw = TRUE))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7266 -0.5739 -0.1323  0.3710  2.9932
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    9.826e+00  1.146e+01   0.858 0.391753
## poly(DT$index, 5, raw = TRUE)1  3.222e-01  2.345e-01   1.374 0.170296
## poly(DT$index, 5, raw = TRUE)2 -4.913e-03  1.831e-03  -2.683 0.007667 **
## poly(DT$index, 5, raw = TRUE)3  2.413e-05  6.849e-06   3.523 0.000486 ***
## poly(DT$index, 5, raw = TRUE)4 -5.019e-08  1.232e-08  -4.075 5.75e-05 ***
## poly(DT$index, 5, raw = TRUE)5  3.785e-11  8.552e-12   4.426 1.30e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8297 on 332 degrees of freedom
##   (118 observations deleted due to missingness)
## Multiple R-squared:  0.9058, Adjusted R-squared:  0.9044
## F-statistic: 638.7 on 5 and 332 DF,  p-value: < 2.2e-16
```

```
# find outliers whose residuals' absolute value larger than 1.5
criterion = 1.5
DT$libor_1M_predict <- predict(fit1b, data.table(x=DT$index))
DT$libor_1M_upper <- DT$libor_1M_predict+criterion
DT$libor_1M_lower <- DT$libor_1M_predict-criterion
DT$libor_1M_residuals <- c(rep(NA, 118), residuals(fit1b))
DT$libor_1M_outlier <- ifelse(abs(DT$libor_1M_residuals)>criterion, DT$libor_1M,NA)
libor_1M_outlier_n <- sum(!is.na(DT$libor_1M_outlier))

# plot Libor_1M and outliers
ggplot(DT,aes(date)) +
  geom_line(aes(y = libor_1M, col = "libor_1M")) +
  geom_line(aes(y = libor_1M_predict, col = "libor_1M_fit")) +
  geom_line(aes(y = libor_1M_upper, col = "Upper Boundary"), linetype = "dashed") +
  geom_line(aes(y = libor_1M_lower, col = "Lower Boundary"), linetype = "dashed") +
  geom_point(aes(y = libor_1M_outlier)) +
  xlim(as.Date(c('1989/11/01', '2017/12/01'), format="%Y/%m/%d")) +
  ggtitle("Figure 1(b) Libor_1M VS Outliers") +
  theme_bw() + xlab("date") + ylab("")
```



From the the above plot of Libor_1M, we can see it is like an exponetial curve with 2 jumps. For simplicity, we use 5-order polynomial regression to fit it. From the regression summary, we can see the 2-5 order parameters are significant and R square reaches 0.9058. Additionally, as shown in the above plot, the regression green line seems to be a good fit for Libor_1M. We define outliers as those whose residuals' absolute value larger than a given criterion, for instance, 1.5. Finally, we find 24 outliers for Libor_1M.

(c) Build a simple quarterly model that explains the VIX index by one or more other variables provided. Provide a measure of how your model performs (in-sample).

VIX index is the implied volatility of S&P 500 index options. Although the best model to simulate heteroscedasticity is GARCH model, we do not need to build VIX in GARCH for this question. Because we can extract volatility from given equity index and then run linear regression of VIX on the extracted volatility. Since S&P 500 is a portfolio index, we regress on both Equity 1 and Equity 2 to improve accuracy.

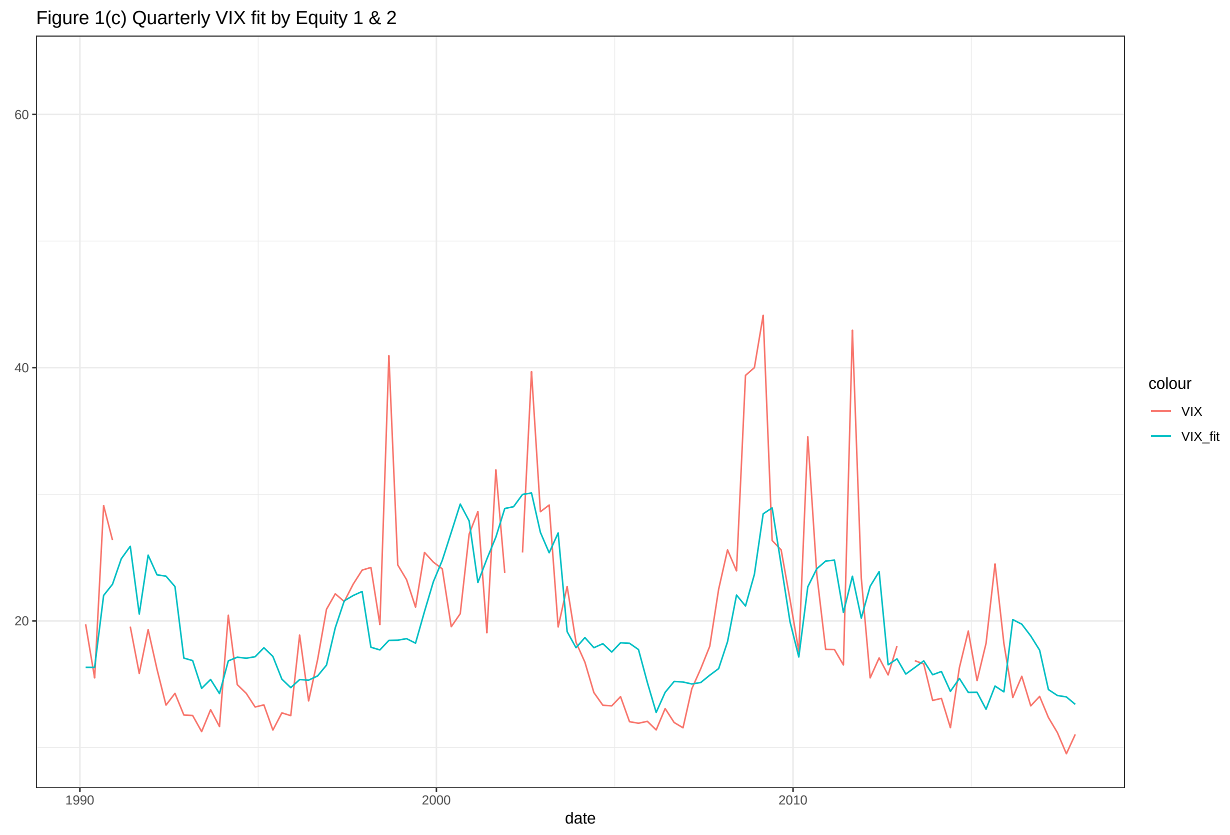
```
library(zoo)

# calculate the rolling volatility with window period of 1 year for Equity 1 & 2 returns
DT_quarter[, Equity1_ret_sd := c(rep(NA,3), rollapply(DT_quarter$Equity1_ret, width=4, FUN = sd))]
DT_quarter[, Equity2_ret_sd := c(rep(NA,3), rollapply(DT_quarter$Equity2_ret, width=4, FUN = sd))]

# run linear regressions of VIX on Equity 1 & 2 rolling volatiliies
fit1c <- lm(VIX ~ Equity1_ret_sd + Equity2_ret_sd, data=DT_quarter)
summary(fit1c)
```

```
##
## Call:
## lm(formula = VIX ~ Equity1_ret_sd + Equity2_ret_sd, data = DT_quarter)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.182  -4.251  -1.050   3.179  22.490
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      11.356      1.216   9.339 1.73e-15 ***
## Equity1_ret_sd  147.046     35.865   4.100 8.12e-05 ***
## Equity2_ret_sd   35.932     11.058   3.250  0.00155 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.073 on 106 degrees of freedom
##   (43 observations deleted due to missingness)
## Multiple R-squared:  0.3545, Adjusted R-squared:  0.3423
## F-statistic:  29.1 on 2 and 106 DF,  p-value: 8.417e-11
```

We calculate rolling volatility for Equity 1 and Equity 2 within the last 1 year. Then we run linear regressions and get the summary as shown above. Estimated parameters are significant since all 3 p-values are small. RSE is 6.073 and R square is 0.3545.

```
# get fitted values of VIX by the linear regression model
DT_quarter$PredictVIX_1c <- predict(fit1c, data.table(Equity1_ret_sd=DT_quarter$Equity1_ret_sd,
                                                      Equity2_ret_sd=DT_quarter$Equity2_ret_sd))
```

```r
# plot VIX and the fitted results
ggplot(DT_quarter,aes(date)) +
  geom_line(aes(y = VIX, col = "VIX")) +
  geom_line(aes(y = DT_quarter$PredictVIX_1c, col = "VIX_fit")) +
  xlim(as.Date(c('1990/03/01', '2017/12/01'), format="%Y/%m/%d")) +
  ggtitle("Figure 1(c) Quarterly VIX fit by Equity 1 & 2") +
  theme_bw() + xlab("date") + ylab("")
```



We also plot the original VIX and fitted VIX as above. We can see for a simple linear regression model, it is a good fit for VIX.

(d) Give an estimate of the VIX index level in a scenario where the Equity Index 1 drops to 1300 at the end of the first quarter of 2018 (i.e., the level of Equity index 1 is 1300 at date 201803). Same question if the Equity Index 2 increases to level 10000 at date 201803.

```r
#calculate the correlation of Equity 1 volatility and Equity 2 volatility
cor(DT_quarter$Equity1_ret_sd, DT_quarter$Equity2_ret_sd, use = "complete.obs")
```

```
## [1] 0.0501094
```

Since in 1(c) we build VIX on both Equity 1 and Equity 2, we first calculate the correlation between the volatilities of the two index returns and get 0.05. Therefore, for simplicity, we can assume changes of the 2 index volatilities are independent.

```r
# calculate the Equity 1 return volatility in 201803 if Equity 1 drops to 1300
# assuming 201801-201802 Equity 1 index drops continuously
Equity1_ret_201803 = log(1300/tail(DT_quarter$Equity1,1))/3
Equity1_ret_sd_201803 <- sd(c(tail(DT_quarter$Equity1_ret,3), Equity1_ret_201803))

# predict VIX assuming Equity 2 volatility stays the same
PredictVIX_1d_1 <- predict(fit1c, data.table(Equity1_ret_sd=Equity1_ret_sd_201803,
                                             Equity2_ret_sd=tail(DT_quarter$Equity2_ret_sd,1)))
unname(PredictVIX_1d_1)
```

```
## [1] 30.70763
```

If Equity 1 drops to 1300 in 201803, we first calculate the corresponding volatility in 201803. Then we predict VIX based on the model in 1(c), assuming Equity 2 volatility stays the same as in 201712. The resulting prediction for 201803 VIX is 30.71.

```r
# calculate the Equity 2 return volatility in 201803 if Equity 2 increases to 10000
Equity2_ret_201803 = log(10000/tail(DT_quarter$Equity2,1))
Equity2_ret_sd_201803 <- sd(c(tail(DT_quarter$Equity2_ret,3), Equity2_ret_201803))

# predict VIX assuming Equity 1 volatility stays the same
PredictVIX_1d_2 <- predict(fit1c, data.table(Equity1_ret_sd=tail(DT_quarter$Equity1_ret_sd,1),
                                             Equity2_ret_sd=Equity2_ret_sd_201803))
unname(PredictVIX_1d_2)
```

```
## [1] 18.31711
```

If Equity 2 increases to 10000 in 201803, we use the same method. The resulting prediction for 201803 VIX is 18.32. We find that Equity 1 has more effect on VIX than Equity 2. This is because for the linear regression model in 1(c), weight of Equity 1 is higher than that of Equity 2.

Question 2: Longest run of heads in a sequence of coin tosses

(a) Write a function which generates random sequences of heads and tails for any value of n.

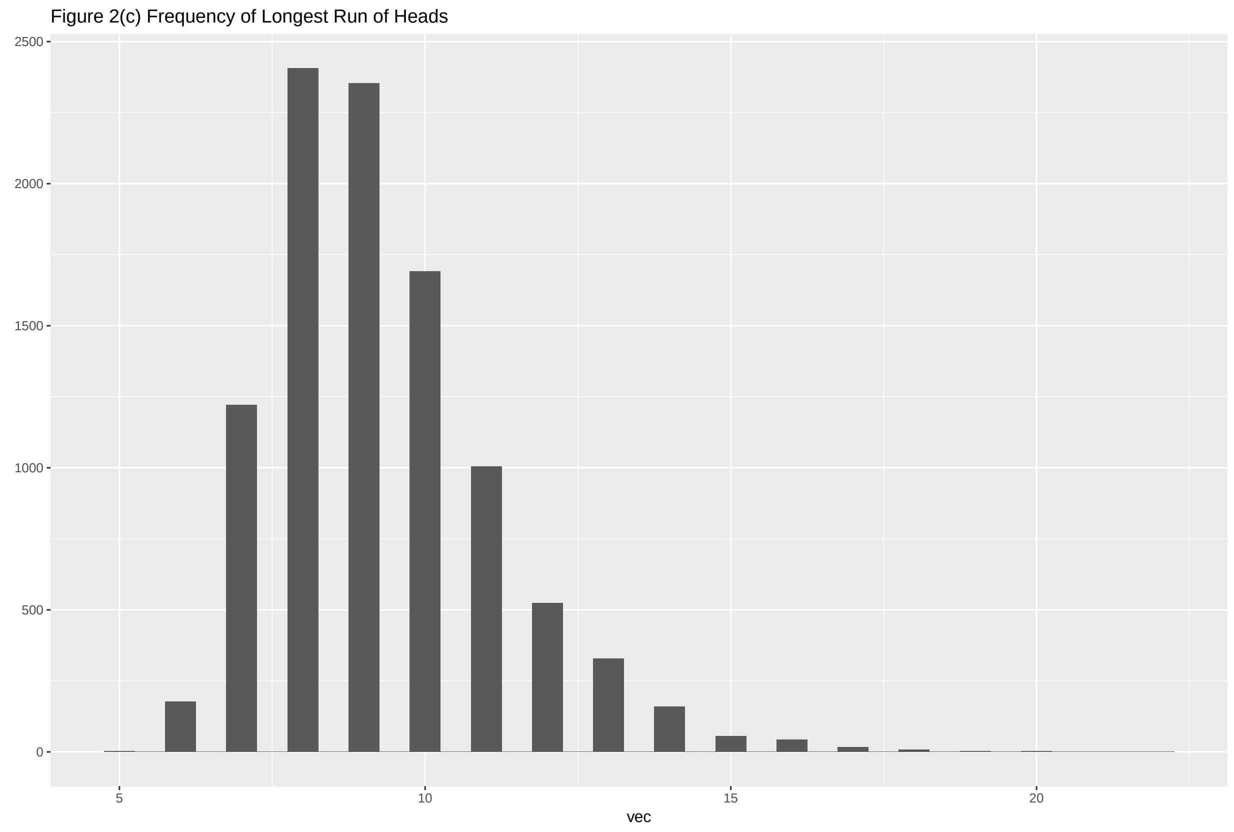
```r
# Use random binomial to generate 1 ("H") and 0 ("T")
rantoss <- function(n){
  res = rbinom(n, 1, 0.5)
  res[res==1] <- "H"
  res[res==0] <- "T"
  return(res)
}
```

(b) Write a function which computes the length of the longest run of heads in an arbitrary sequence of heads and tails.

```r
# Loop over sequence generated by (a) function to get the longest run of heads
LonestH <- function(n){
  seq = rantoss(n)
  res = 0
  count = 0
  for(i in 1:n){
    if (seq[i]=="H") {
      count = count + 1
    } else {
      if(count>res){
        res = count
      }
      count = 0
    }
  }
  return(res)
}
```

(c) For a sequence of length $n = 1000$, we have observed a longest run of heads equal to $M_n = 6$. Based on this piece of information, do you believe that the coin is fair?

```
# simulate 10000 times for sequence with length 1000
sim = 10000
vec <- rep(0, sim)
for(i in 1:sim){
  vec[i]=LonestH(1000)
}

# plot the histogram for frequency
qplot(vec, geom="histogram", main="Figure 2(c) Frequency of Longest Run of Heads",
      binwidth = 0.5)
```



```r
# calculate the t value
M_n = 6
t_star <- (mean(vec)- M_n)/(sd(vec)/sqrt(sim))
t_value <- qt(0.05/2, sim)
```

We simulate 10,000 times for sequence with length of 1,000 and plot the histogram of longest run of heads. We can see the probability of $M_n = 6$ is quite small. Statistically, we estimate $\tilde{M_n}$ by calculating its average and then apply t-test with null hypothesis: $\tilde{M_n} = 6$. We get t-statistic $\frac{mean(M_n)-6}{std(M_n)/\sqrt{N}} = 177$, much higher than the 95% confidence interval t-value 1.96. Thus we reject the null hypothesis that it is a fair coin with 95% confidence interval.