Question 3

Sun Yutong (Main) Tang Chen (Review) Ye Chen (Review)

2023-10-26

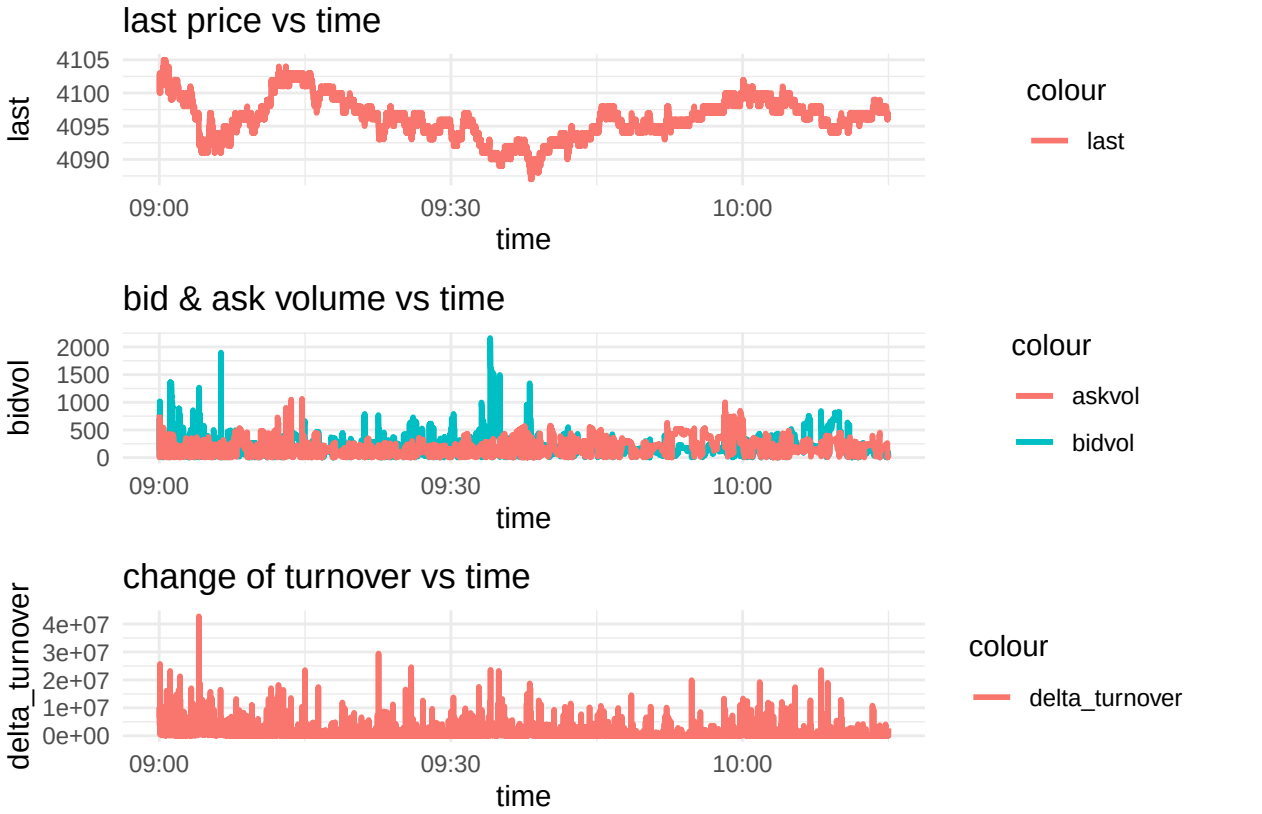
(Most codes will not be shown in the report. Please refer to rmd file and Appendix for code block and less important graphs)

1. Explanatory Data Analysis of Original Data

In this section I am going to explore the characteristics of the data set. And data are mutated and stored to multiple e tibbles.

First, look at the time interval in this data. Graph(a) in Appendix verify the most time interval is 0.5 seconds, with small amount of intervals of 1 second and rare occurrence of 1.5/2 seconds intervals. Then let's view the plot of last price and plot of bid & ask volume, and delta_turnover (basically proportional to volume).

```
last_plot/bidask_volume_plot/d_turnover_plot
```



There are interesting findings by comparing the pattern of these 3 figures. First, the last price is moving down and up rapidly and the end position is lower than the initial position. It suggests that without high frequency trading, the period of price increase is short and it might be hard for investors to make profit in this period. For the bid & ask volume plot, bid volume is more volatile and there are large volumes, when the

price is dropping. Similarly, even ask volume is more stable but when the price has increase for a while, the ask volume will increase. It demonstrated a significant correlation between the first two plots and indicates the investors' reaction of price change. For the last graph, change of turnover, overall it is volatile and large amount of turnover occur in smaller time intervals.

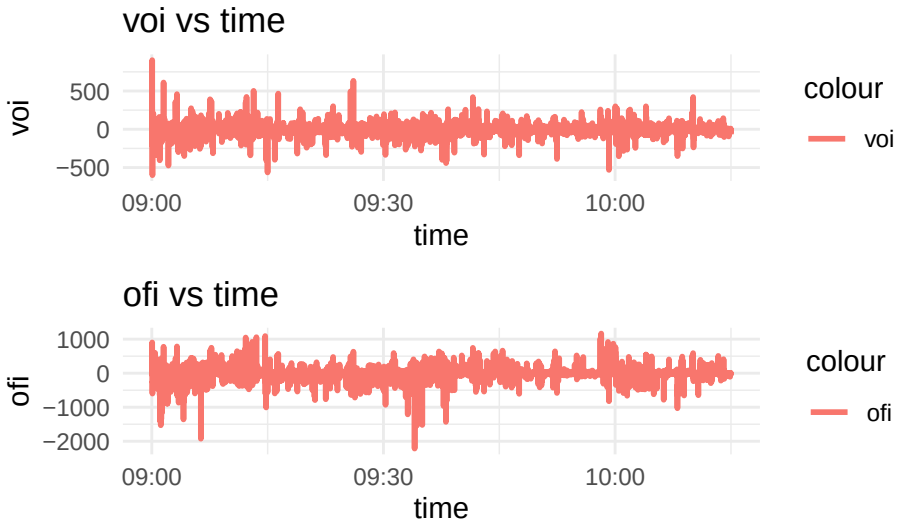
Lets focus more on the extreme values of these attributes. Using code (a) in Appendix, we obtain conclusions that the at 09:00:26 it reaches its maximum price, at 4105, and at 09:38:12 it reaches its minimum price, at 4087, which is after the maximum price time, so we have its maximum drawdown directly: 0.438%. The largest increase of turnover and volume, which are 21866459170 and 533140 respectively, occurs at the same time: 10:15:00.

2. Factor Implementation and associated Explanatory Data Analysis

Then we will implement all factors into a new tibble, factors. We added 2.2 factor, volume imbalance. By reading papers, we add the 2.3 factor, VOI, 2.4 factor, OFI. We also create two more functions to find lagged value, and rolling sum with a customized interval. Then we define the 5 & 10 interval price change as the difference of last price at the start and end of interval, and define 5 & 10 intereval VOI, OFI as VOI_5, VOI_10, OFI_5, OFI_10 by calculating their rolling sum, as advised by paper.

Then we conduct EDA of VOI and OFI. By comparing the voi and ofi plot, we can see the change of these 2 factors are overall volatile and the pattern has significant similarities, while ofi has larger value. We have also checked the VOI_10 and OFI_10 has similar pattern.

```
voi_plot/ofi_plot
```



Referring to Graph Block (b), we have plotted the autocorrelation function of voi, ofi and their change. We have re-produce the ACF figures for VOI factors.It is obvious that the autocorrelation of voi is consistent with the plots generated in page 6 of hftois paper, and ofi demonstrated simimlar properties, means the factors has positive autocorrelation and change has negative autocorrelation.

3. Testing Factor usefulness

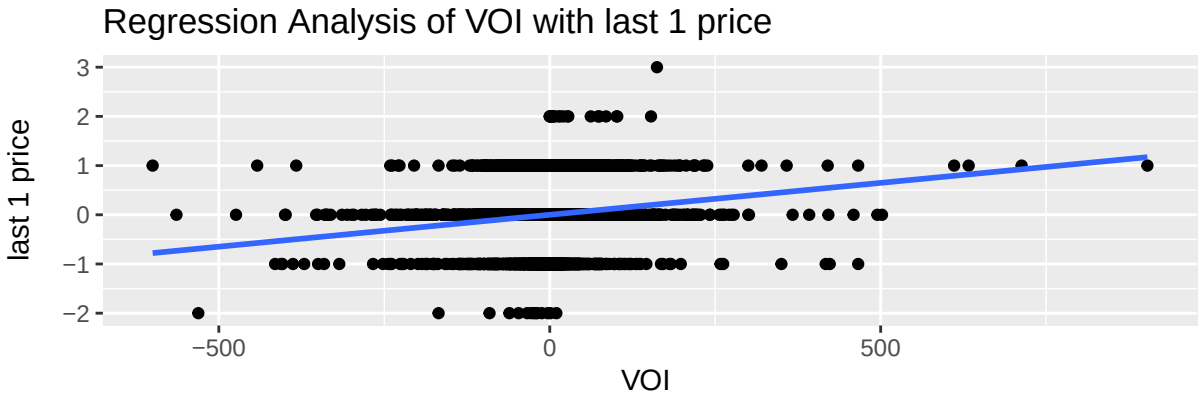
Then, we will test various factors, using regression and test statistics. We will run regression, using current price change as dependent variable for voi and ofi, and using last 10 interval change as dependent variable for voi_10 and ofi_10.

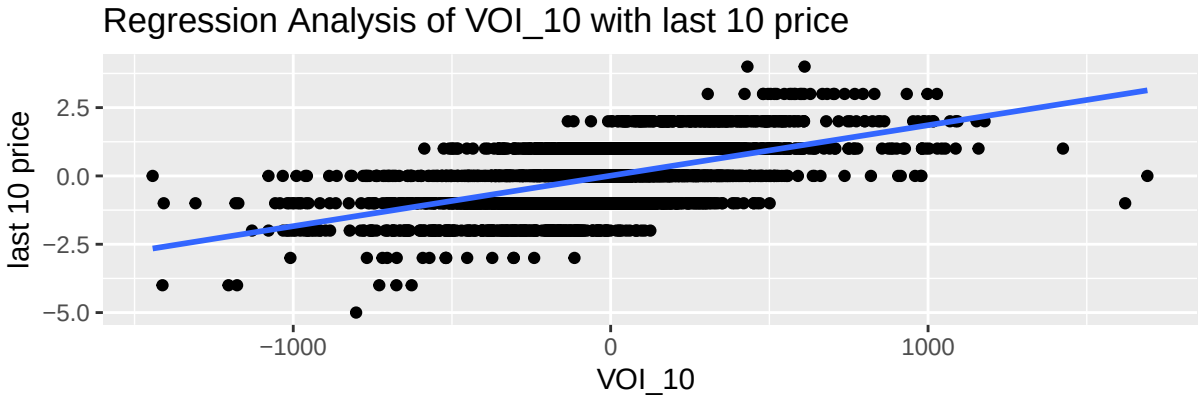
```
summary(linear_model_voi)
```

```
##
```

```
## Call:
## lm(formula = last_1 ~ voi, data = factors)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.01312 -0.03388 -0.00015  0.03358  2.78970
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0001504  0.0063455   0.024    0.981
## voi         0.0012972  0.0001196  10.842   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5974 on 8863 degrees of freedom
## Multiple R-squared:  0.01309,    Adjusted R-squared:  0.01298
## F-statistic: 117.6 on 1 and 8863 DF,  p-value: < 2.2e-16
```

```
voi_plot/voi_10_plot
```





We demonstrate the summary of model for voi vs last_1 price regression here, we can see the p value is small, rejecting the hypothesis that the factor loading of factor voi is small. Also, the plot of regression for voi and voi_10 are displayed, suggesting strong, positive linear relatipnship. The summary of voi_10 vslast_10 model and same procedure for ofi factor are displayed in appendix, code (b), and the result are consistent with ofi. In summary, we can see both voi and ofi are significant, and the two factors demonstrated great similarities, and we have successfully reproduced the test result in the hftois paper for voi factor.

Also, referring to code (c) we have test other factor, such as micro price - middle price, but regression testing has failed, with a p-value greater than 0.1. It means current the useful factors are the voi and ofi concept

factors.

4. Back-Testing of Factor and Calculate Return

Here we will propose a simple, high frequency strategy try to make a profit in this morning. It is just a simply back testing with one factor, OFI_10. Considering we only have OFI and VOI concept factors, and we have not tested for multiple factor regressions, we can only have a 1 factor model, otherwise it will lead to serious multilinearity problems. Also, we cannot conduct research to optimise the buy/sell signal value or which factor is more appropriate, since we only have 1 asset and limited data, so this back-testing is just a simple demo that, with a robust setting, we can use high frequency trading to make a profit even the asset price trend is downwards.

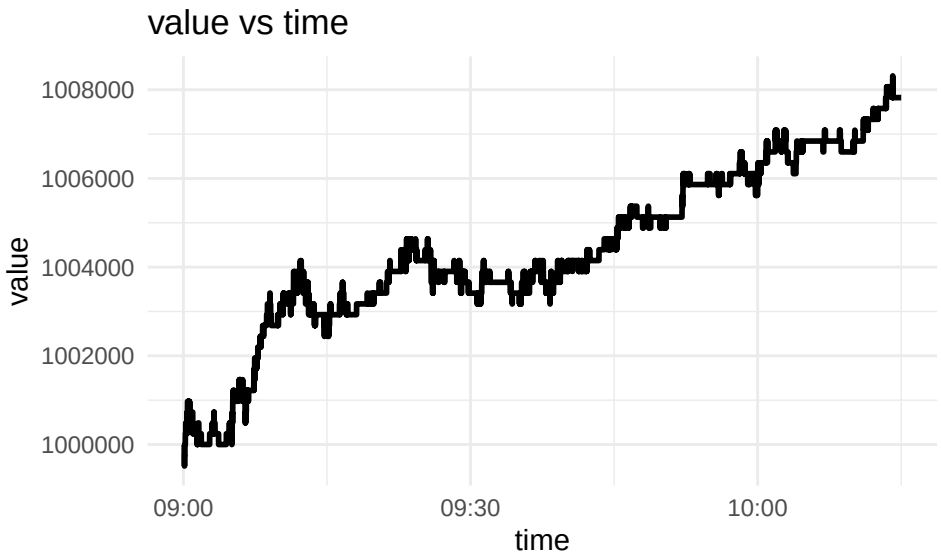
The rules of strategy are: 1. No transaction costs 2. We are only price takers 3. Initial wealth $100,0000 4. Once the sell/buy signal appears, spend all funds on selling, buying. 5. When holding for more than 10 intervals (approximate 5 seconds), sell immediately. 6. Buy signal is triggered when previous OFI_10 is greater than its 0.8 quantile, and sell signal is triggered when previous OFI_10 is smaller than its 0.2 quantile.

```
#  Back-testing with OFI_10 factor
sell_val = quantile(factors$ofi_10, 0.2) # value -232
buy_val = quantile(factors$ofi_10, 0.8) # value 165
# initial saving
account <- 1000000
# initialize buy & sell signal
buy_signal <- 0
sell_signal <- 0
count <- 0
holding <- 0
buy_signal <- 0
sell_signal <- 0
curren_val <- c()
for (i in 2:length(hfdata$time)) {
  count = count + 1
  # holding for over 5 seconds
  if (sell_signal == 0 && count > 10) {
    sell_signal <- 1
    buy_signal <- 0
    count <- 0
  }
  # seen buy signal
  if ((factors$ofi_10[i-1] >= buy_val)) {
    buy_signal <- 1
    sell_signal <- 0
  }
  # seen sell signal
  if ((factors$ofi_10[i-1] <= sell_val)) {
    buy_signal <- 0
    sell_signal <- 1
  }
  # in the end, sell everything
  if (i == length(hfdata$time)) {
    #print(hfdata$time[i])
    sell_signal <- 1
  }
```

```
  # perform sell
  if (sell_signal == 1 && holding != 0) {
    account <- account +  hfdata$last[i] * holding
    holding <- 0
    sell_signal <- 0
    buy_signal <- 0
  }
  # perform buy
  else if (buy_signal == 1 && holding == 0)  {
    holding <- account %/% hfdata$last[i]
    account <- account %% hfdata$last[i]
    buy_signal = 0
  }
  curren_val <- c(curren_val, holding*hfdata$last[i] + account)
}
# save value for plotting
plot_account <- tibble(
  value = curren_val,
  time = hfdata$time[2:length(hfdata$time)]
)
print(account)
```

```
## [1] 1007823
```

```
value_plot
```



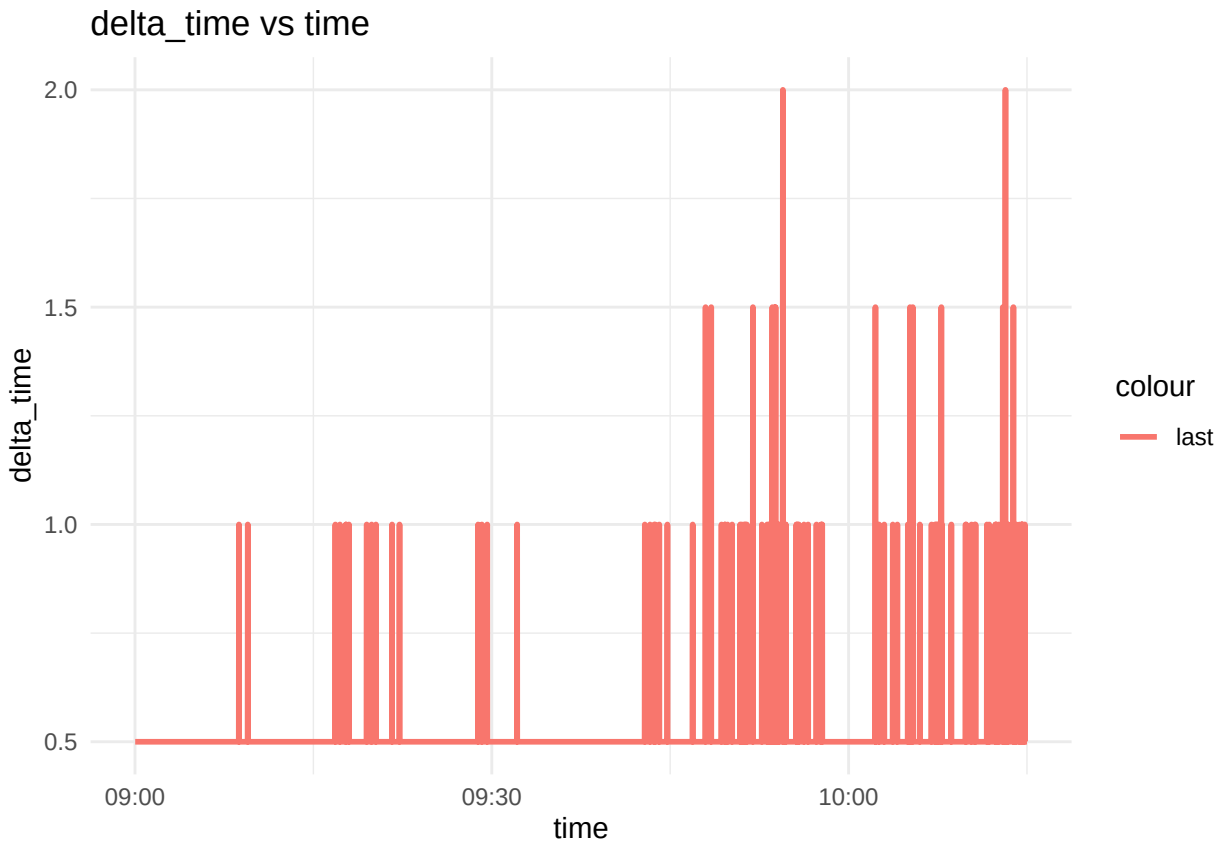
By checking with the plot of value vs time, we can see the strategy is making profit constantly, finding opportunities with the OFI_10 factors and accumulates wealth step by step. In the end the account value is 1007823, suggesting it makes a 0.78% return in 90 minutes, when the actual price movement is -0.12%, suggesting our strategy and factor is useful. Nevertheless, in real life scenario, we still need to optimise this strategy by improving the signal bound, and consider transaction cost, that is important in high frequency trading.

Appendix for Question 3

Graph (a)

```
time_interval_plot
```

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.
```



Code (a)

```
max(hfdata$last)
```

```
## [1] 4105
```

```
min(hfdata$last)
```

```
## [1] 4087
```

```
hfdata$time[which.max(hfdata$last)]
```

```
## [1] "2023-01-10 09:00:26 +08"
```

```
hfdata$time[which.min(hfdata$last)]
```

```
## [1] "2023-01-10 09:38:12 +08"
```

```
index_of_min <- which.min(hfdata$last)
max_drawdown <- (max(hfdata$last) - min(hfdata$last))/max(hfdata$last)

max(hfdata$turnover)
```

```
## [1] 21866459170
```

```
max(hfdata$volume)
```

```
## [1] 533140
```

```
hfdata$time[which.max(hfdata$turnover)]
```

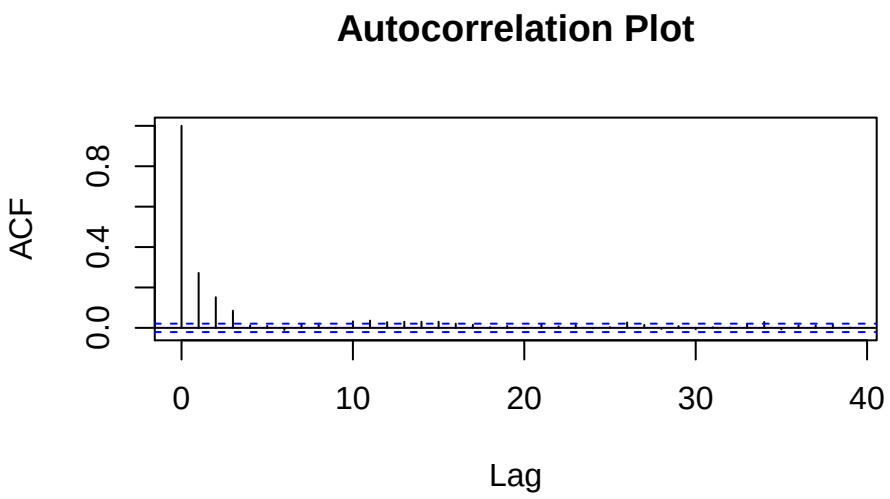
```
## [1] "2023-01-10 10:15:00 +08"
```

```
hfdata$time[which.max(hfdata$volume)]
```

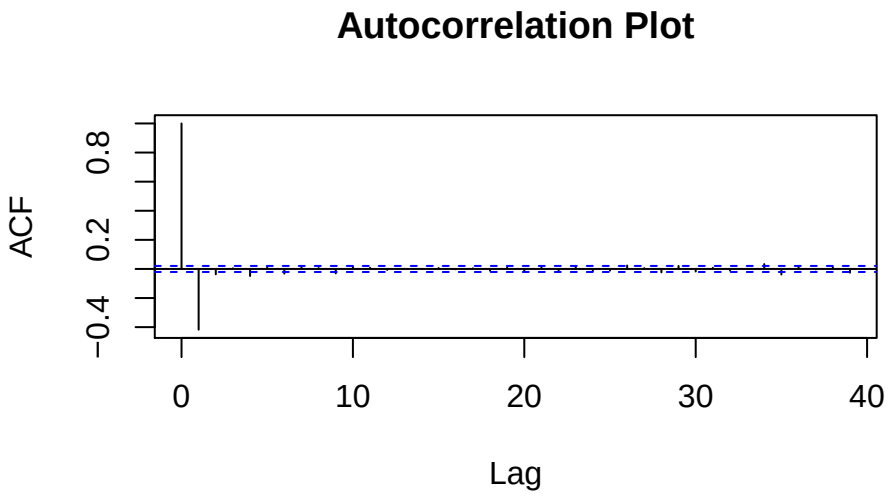
```
## [1] "2023-01-10 10:15:00 +08"
```

Graph Block (b)

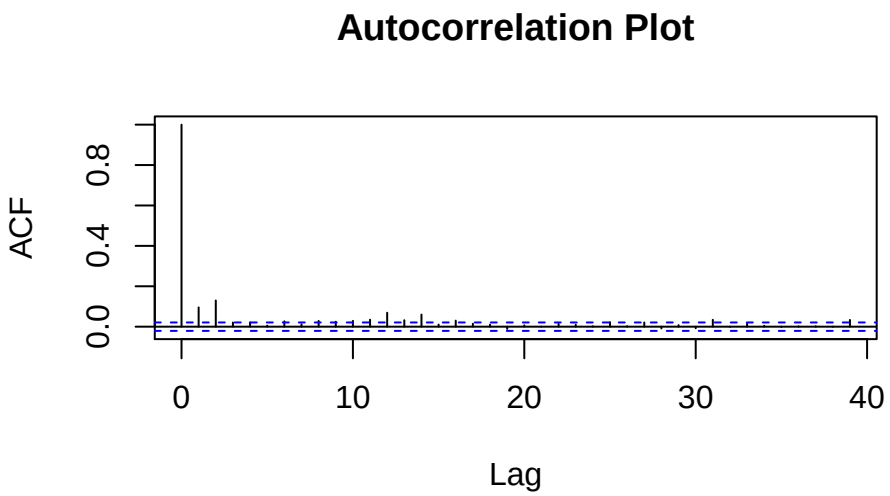
```
a1 <- acf(factors$voi, main="Autocorrelation Plot")
```



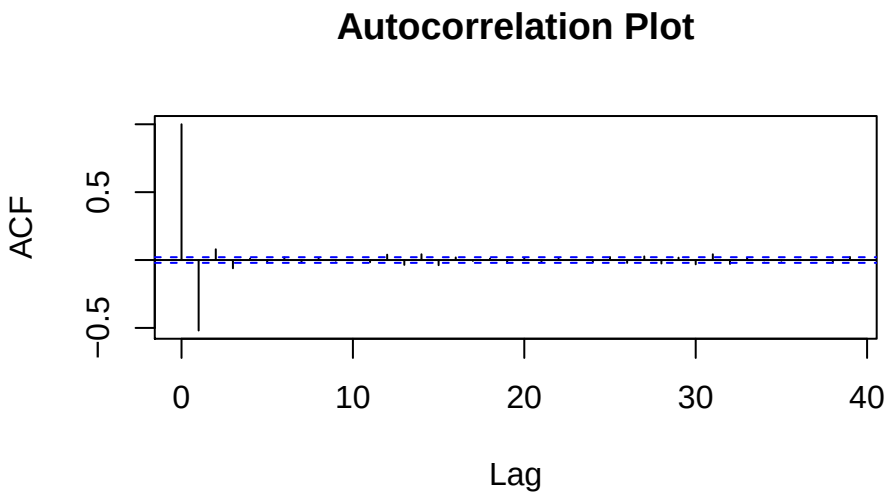

```
a2 <- acf(diff(factors$voi), main="Autocorrelation Plot")
```



```r
a3 <- acf(factors$ofi, main="Autocorrelation Plot")
```




```r
a4 <- acf(diff(factors$ofi), main="Autocorrelation Plot")
```



Code (b)

```r
summary(linear_model_voi_10)
```

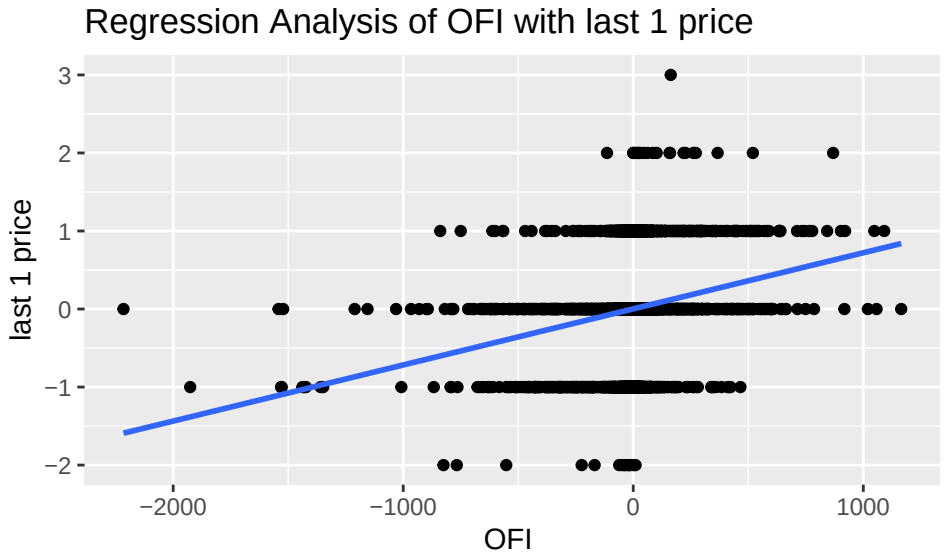
```
##
## Call:
## lm(formula = last_10 ~ voi_10, data = factors)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0029 -0.4928  0.0003  0.5101  3.1950
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.937e-03  7.999e-03   1.117    0.264
## voi_10      1.847e-03  3.566e-05  51.798   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.7526 on 8863 degrees of freedom
## Multiple R-squared:  0.2324, Adjusted R-squared:  0.2323
## F-statistic:  2683 on 1 and 8863 DF,  p-value: < 2.2e-16
```

```
# Perform linear regression
linear_model <- lm(last_1 ~ ofi, data = factors)
# Summary of the model
summary(linear_model)
```

```
##
## Call:
## lm(formula = last_1 ~ ofi, data = factors)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.00909 -0.02275 -0.00190  0.02038  2.88092
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.902e-03  6.312e-03   0.301    0.763
## ofi         7.189e-04  4.853e-05  14.814   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.594 on 8863 degrees of freedom
## Multiple R-squared:  0.02416,    Adjusted R-squared:  0.02405
## F-statistic: 219.5 on 1 and 8863 DF,  p-value: < 2.2e-16
```

```
# Create a scatter plot with the regression line
ggplot(factors, aes(x = ofi, y = last_1)) +
  geom_point() +  # Plot the original data points
  geom_smooth(method = 'lm', formula = y ~ x, se = FALSE) +  # Add the regression line
  labs(title = "Regression Analysis of OFI with last 1 price", x = "OFI ", y = "last 1 price")
```

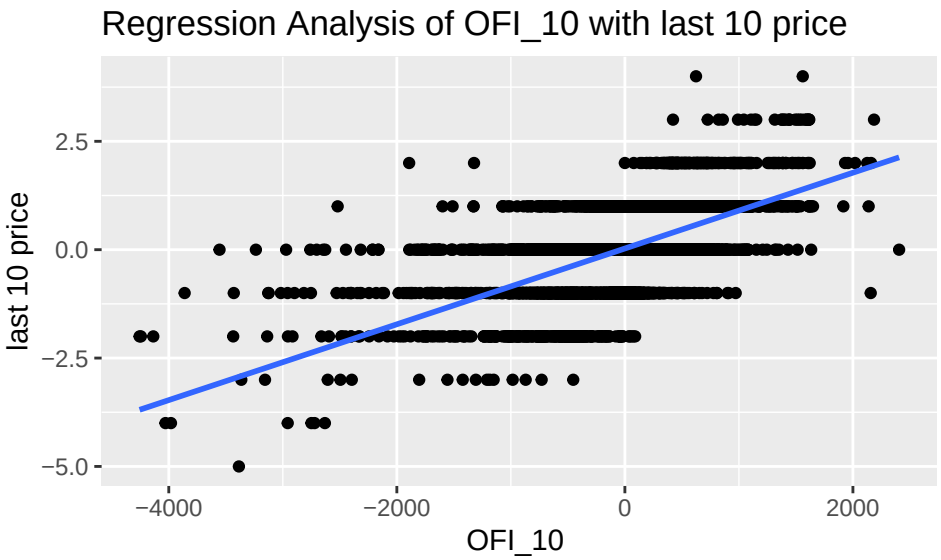



```
# Perform linear regression
linear_model <- lm(last_10 ~ ofi_10, data = factors)
# Summary of the model
```

```r
summary(linear_model)
```

```
##
## Call:
## lm(formula = last_10 ~ ofi_10, data = factors)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9123 -0.4906 -0.0222  0.5216  3.6267
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.741e-02  7.899e-03    3.47 0.000523 ***
## ofi_10      8.743e-04  1.587e-05   55.08  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7414 on 8863 degrees of freedom
## Multiple R-squared:  0.255,  Adjusted R-squared:  0.2549
## F-statistic:  3034 on 1 and 8863 DF,  p-value: < 2.2e-16
```

```r
# Create a scatter plot with the regression line
ggplot(factors, aes(x = ofi_10, y = last_10)) +
  geom_point() +  # Plot the original data points
  geom_smooth(method = 'lm', formula = y ~ x, se = FALSE) +  # Add the regression line
  labs(title = "Regression Analysis of OFI_10 with last 10 price", x = "OFI_10 ", y = "last 10 price")
```



Code (c)

```r
# Perform linear regression
linear_model <- lm(last_1 ~ (micro_price-mid_price), data = factors)
# Summary of the model
summary(linear_model)
```

```
##
## Call:
```

```
## lm(formula = last_1 ~ (micro_price - mid_price), data = factors)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.01821 -0.01053  0.00040  0.01243  2.98245
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.952326   8.414401  -1.539    0.124
## micro_price   0.003162   0.002054   1.539    0.124
##
## Residual standard error: 0.6013 on 8863 degrees of freedom
## Multiple R-squared:  0.0002672,  Adjusted R-squared:  0.0001544
## F-statistic: 2.369 on 1 and 8863 DF,  p-value: 0.1238
```

```r
# Create a scatter plot with the regression line
ggplot(factors, aes(x = (micro_price-mid_price), y = last_1)) +
  geom_point() +  # Plot the original data points
  geom_smooth(method = 'lm', formula = y ~ x, se = FALSE) +  # Add the regression line
  labs(title = "Regression Analysis of micro_price-mid_price with last 1 price", x = "micro_price-mid_pr
```