Analysis of Shanghai Index (Optional)

(Jing Li, Miami University)

1. This note illustrates how to conduct a preliminary statistical analysis of Shanghai Stock Exchange Composite Index (Shanghai Index for short). The one-year data in csv format is downloaded at marketwatch.com. I save the csv file in my dropbox folder. Commands below set working directory and read data into R

```
> setwd("/Users/lij14/Dropbox")
> data =  read.csv("515_sh_index.csv")
> head(data)
        Date     Open     High      Low    Close
1 10/18/2024 3,165.97 3,313.98 3,152.82 3,261.56
2 10/17/2024 3,220.73 3,241.57 3,169.38 3,169.38
3 10/16/2024 3,169.17 3,236.85 3,167.74 3,202.95
4 10/15/2024 3,265.75 3,285.21 3,201.29 3,201.29
5 10/14/2024 3,241.43 3,294.62 3,203.90 3,284.32
6 10/11/2024 3,287.87 3,297.75 3,187.99 3,217.74
>  str(data)
'data.frame':        241 obs. of  5 variables:
 $ Date : chr  "10/18/2024" "10/17/2024" "10/16/2024" "10/15/2024" ...
 $ Open : chr  "3,165.97" "3,220.73" "3,169.17" "3,265.75" ...
 $ High : chr  "3,313.98" "3,241.57" "3,236.85" "3,285.21" ...
 $ Low  : chr  "3,152.82" "3,169.38" "3,167.74" "3,201.29" ...
 $ Close: chr  "3,261.56" "3,169.38" "3,202.95" "3,201.29" ...
```

2. This is an example of "dirty data" that needs cleaning. Even though the data looks fine, the str function shows

   (a) Date is character (chr) because of slash between month, day and year

   (b) Open, High, Low and Close are all chr because of comma

   It is important to check the data format. R cannot report mean or run regression with chr data.

3. Next we convert data into date and numeric format so that R can analyze. We focus on date and close only, and store them in a data frame.

```
> date = as.Date(data$Date, format = "%m/%d/%Y")
> close = as.numeric(gsub(",", "", data$Close))
> d = data.frame(date,close)
> head(d)
        date   close
1 2024-10-18 3261.56
2 2024-10-17 3169.38
3 2024-10-16 3202.95
4 2024-10-15 3201.29
5 2024-10-14 3284.32
6 2024-10-11 3217.74
> str(d)
'data.frame':        241 obs. of  2 variables:
 $ date : Date, format: "2024-10-18" "2024-10-17" ...
 $ close: num  3262 3169 3203 3201 3284 ...
```

Lesson: to figure out those codes, you may ask chatgpt "How to convert string 10/17/2024 into date in R", or "how to convert string 3,220.73 into numeric". Other than using R, you can convert data into suitable format in Excel.

4. We prefer that date be in ascending order (the last observation is the latest). So we use order function to sort the data by date. Then we generate the first lag value of close price, and compute the daily return (percent) as

$$return_t = 100\left(\frac{close_t - close_{t-1}}{close_{t-1}}\right) \tag{1}$$

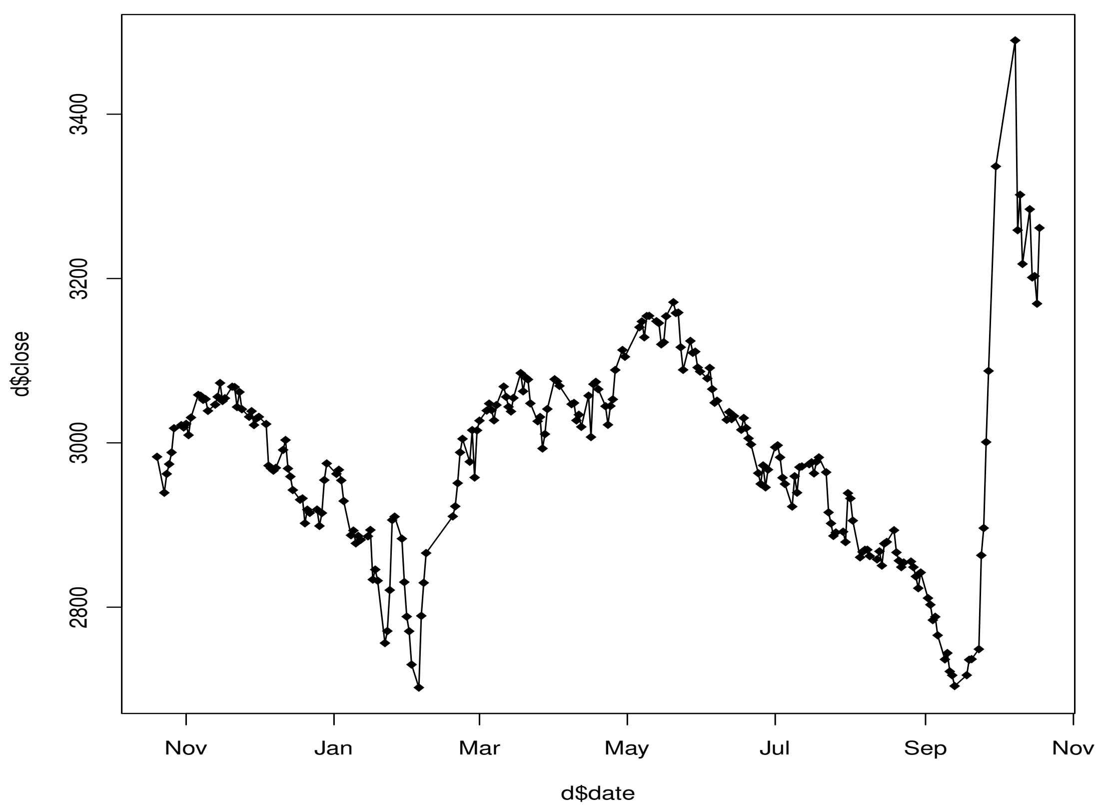
```
> d = d[order(d$date),]
> n = nrow(d)
> d$close.lag = c(NA, d$close[1:(n-1)])
> d$return = (d$close-d$close.lag)/d$close.lag*100
> tail(d)
        date   close close.lag     return
6 2024-10-11 3217.74   3301.93 -2.5497209
5 2024-10-14 3284.32   3217.74  2.0691541
4 2024-10-15 3201.29   3284.32 -2.5280728
```

```
3 2024-10-16 3202.95   3201.29  0.0518541
2 2024-10-17 3169.38   3202.95 -1.0480963
1 2024-10-18 3261.56   3169.38  2.9084553
```

Notice that missing value NA is created for the lag close and return. We see, for instance, as the close price rises from 3169.38 to 3261.56, the daily return is about 2.9 percent on October 18, 2024.

5. It is a good idea to start data analysis with a plot of close price

```
plot(d$date,d$close,type="o",pch=18)
```



We see a big jump from below 2800 to above 3400 in mid September, which is driven by an unprecedented market intervention by Chinese government—big firms are provided bank loan to repurchase their stocks.

6. We are dealing with time series data here. In general stock price today depends on yesterday, so data are dependent and the i.i.d assumption fails. In fact, for time series

data a critical issue is unit root or nonstationarity (Google it!). So next we perform the Augmented Dickey-Fuller (ADF) unit root test with the adf.test function in the tseries package

```
> library(tseries)
> adf.test(d$close)

        Augmented Dickey-Fuller Test

data:  d$close
Dickey-Fuller = -1.8537, Lag order = 6, p-value = 0.6374
alternative hypothesis: stationary
> adf.test(d$close, alternative = c("explosive"))

        Augmented Dickey-Fuller Test

data:  d$close
Dickey-Fuller = -1.8537, Lag order = 6, p-value = 0.3626
alternative hypothesis: explosive
```

The null hypothesis is that the close price is nonstationary (have unit root). We find that the null hypothesis cannot be rejected no matter the testing regression includes a trend (p-value is 0.3626) or no trend (p value is 0.6374).

7. The finding of unit root in the close price implies that many classical results you learn in statistics course become invalid since they assume data are independent. For instance, consider computing the sample mean of close price and testing the hypothesis of average close price being 3000. R codes are below:

```
> mean(d$close)
[1] 2981.545
> se = sd(d$close)/sqrt(length(d$close))
> se
[1] 7.86105
> t = (mean(d$close)-3000)/se
> t
```

```
[1] -2.347706
> t.test(d$close,mu=3000)

        One Sample t-test

data:  d$close
t = -2.3477, df = 240, p-value = 0.0197
alternative hypothesis: true mean is not equal to 3000
95 percent confidence interval: 2966.059 2997.030
sample estimates:
mean of x
 2981.545
```

(a) First of all, the sample mean 2981.545 is irrelevant because a time series with unit root has no tendency to revert to the mean. In other words, we cannot expect that in long run the close price will move toward 2981.545. In fact, we see in the time series plot that from June to September the close price seems to move further and further away from 3000 with no tendency to rebound. In an upper level time series class you will learn that a nonstationary time series has time-varying mean, so reporting just one sample mean for the whole sample is very misleading.

(b) Second, the traditional formula for standard error is wrong since it assumes data are independent. In general, for dependent data we can show

$$var(\bar{y}) \neq \frac{\sigma^2}{n}, \quad or \quad se \neq \frac{\sigma}{\sqrt{n}} \tag{2}$$

As a result, the t value -2.3477 is incorrect because it uses the incorrect standard error 7.86105 that underestimates the variation of sample mean by ignoring positive covariances of y_t and its first lag y_{t-1}, second lag y_{t-2}, and so on. We can use math to show the correct se is greater than 10.96734

$$var(\bar{y}) > \frac{var(y_t)}{n} + \frac{cov(y_t, y_{t-1})}{n} \Rightarrow se > 10.96734 \tag{3}$$

```
> sqrt(var(d$close)+cov(d$close,d$close.lag,use="complete.obs"))/sqrt(length(d$
[1] 10.96734
> (mean(d$close)-3000)/10.96
```

```
[1] -1.68389
```

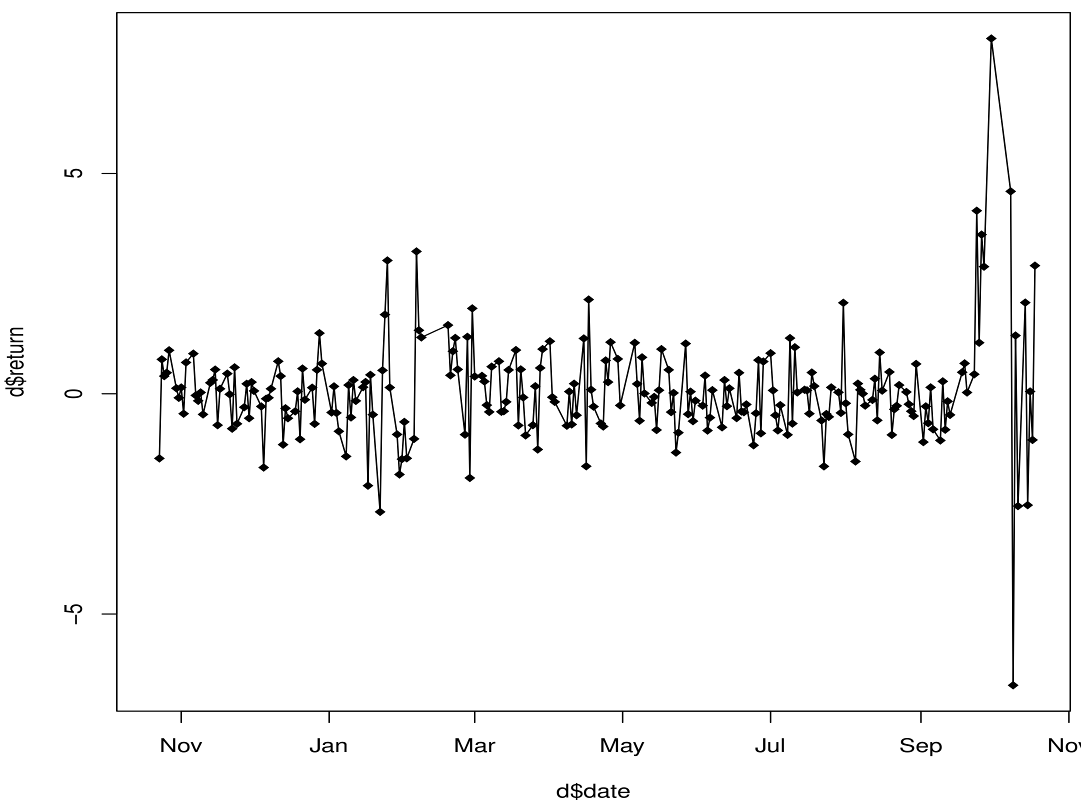
The t value would become -1.68389 if the se=10.96734 was used.

(c) Moreover, the p value 0.0197 and 95 percent confidence interval reported by t.test are all wrong due to the incorrect se and t value.

(d) To make it worse, R gives no warning that those results are wrong!

Lesson: use extreme caution when conducting statistical analysis to time series data with unit root. R results could be wrong, but R gives no warning. A quick and easy fix is avoiding statistical analysis of time series data with unit root.

8. A common way to remove unit root is taking difference, something like formula (1). That means the statistical analysis of return should be more theoretically sound than analyzing close price. The time series of return is below



Now we spot two signals for stationarity or absence of unit root

(a) The return is mean-reverting—it fluctuates frequently around 0

(b) The return shows no local trend or long swing like the one we observe in the close price from June to September

Thanks to a small p value, the ADF test applied to return leads to rejection of unit root

```
> head(d$return)
[1]          NA -1.4672853  0.7808008  0.4007103  0.4771175  0.9865141
> adf.test(d$return[-1])

        Augmented Dickey-Fuller Test

data:  d$return[-1]
Dickey-Fuller = -6.4654, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

Notice that the first observation of return is missing value NA (because lag close price has NA). So we exclude it when performing the ADF test.

9. Lesson: pay attention to missing value when analyzing time series. R cannot handle missing value automatically.

10. Next we report mean, standard deviation, min, max, 10-th percentile and 90-th percentile of return. This time we use na.rm = TRUE to take care of missing value.

```
> mean(d$return, na.rm = TRUE)
[1] 0.0444556
> sd(d$return, na.rm = TRUE)
[1] 1.212394
> min(d$return, na.rm = TRUE)
[1] -6.617036
> max(d$return, na.rm = TRUE)
[1] 8.063727
> quantile(d$return, probs = 0.10, na.rm = TRUE)
-0.9529503
> quantile(d$return, probs = 0.90, na.rm = TRUE)
1.160053
```

```
> t.test(d$return[-1])

        One Sample t-test

data:  d$return[-1]
t = 0.56805, df = 239, p-value = 0.5705
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
-0.1097113  0.1986225
sample estimates:
mean of x
0.0444556
```

(a) Because return is stationary (having no unit root), the sample mean 0.0444556 is meaningful. We conclude that, based on this sample, the average daily return is about 0.04 percent.

(b) The result of t test is likely to be correct: t = 0.56805 and p-value = 0.5705 indicate that we cannot reject hypothesis that the true average daily return be 0. Note that the 95 percent confidence interval (−0.10971130.1986225) contains 0.

11. According to a theory called Efficient Market Hypothesis (EMH), no historical pattern can be used to predict future return. To check that, we consider fitting the first order autoregression AR(1) to return

$$return_t = \beta_0 + \beta_1 return_{t-1} + u_t \qquad (4)$$

The results are below

```
> arima(d$return,order=c(1,0,0))

Coefficients:
         ar1  intercept
      0.0987     0.0451
s.e.  0.0651     0.0862

sigma^2 estimated as 1.45:  log likelihood = -385.12,  aic = 776.24
```

The t value of the slope coefficient is $t_{\beta_1} = \frac{0.0987}{0.0651} < 1.96$. So the first lag return is indeed useless to predict return.

12. How about we run a second order autoregression? The regression is

$$return_t = \beta_0 + \beta_1 return_{t-1} + \beta_2 return_{t-2} + u_t \tag{5}$$

```
> arima(d$return,order=c(2,0,0))
Coefficients:
         ar1     ar2  intercept
      0.0820  0.1500     0.0459
s.e.  0.0647  0.0648     0.0999

sigma^2 estimated as 1.418:  log likelihood = -382.47,  aic = 772.95
```

It seems that for this sample the second lag has some predictive power with t value $t_{\beta_2} = \frac{0.1500}{0.0648} > 1.96$.

13. DO NOT take this particular finding seriously. EMH asserts that when we change the sample, most likely the second lag will be statistically insignificant. In other words, a particular regression may capture noise other than signal (suffering the overfitting issue).

14. You may search internet and learn more about EHM and its implications for the so called technical analysis in particular. This is one reply from chatgpt for this issue—"Under the strictest interpretation of EMH, technical analysis is seen as ineffective because any potential advantages are quickly neutralized by market participants. However, the debate continues, particularly with the emergence of behavioral finance, suggesting that market inefficiencies do exist. This opens the door for technical analysis, as patterns may reflect investor sentiment and behavior." You may read

`https://www.aeaweb.org/articles?id=10.1257/089533003321164958`
`https://www.aeaweb.org/articles?id=10.1257/089533003321164967`

15. In addition to be at odds with EHM, some published "successful investment strategy" may be subject to the overfitting issue, with which I am especially concerned. Many of those strategies are based on <u>statistical analysis of historical</u> data, so I will use

regression as an illustration of the overfitting issue — I will create two variables y and x that are independent of each other. However, when I run a regression that uses x to predict y, about 5 out of 100 times the regression indicates that x has predictive power, despite that by construction x has no bearing on y. Below are codes of running a Monte Carlo simulation

```
n = 100; count = 0
for (i in 1:100) {
        y = rnorm(n)
        x = rnorm(n)
        m = summary(lm(y~x))
        count = count + (abs(m$coef[2,3])>1.96)
}
count
[1] 5
```

If you are familiar with hypothesis testing, you may realize this simulation demonstrates nothing but the type-I error. But in the literature of investment, the type-I error can be translated to publication bias—those 5 regressions with significant t value that implies some "successful strategies" are more likely to get published than other 95 regressions with insignificant t values.

16. Lesson: investment strategy derived from statistical analysis may be subject to type-I error or overfitting. They may capture noise other than signal. Even though they may work well for historical data, there is no guarantee that its success will be repeated when being applied to out-of-sample data. The only promised winner of technical analysis is the person that charges you the transaction fee!