Problem 1 (13 credits)

HW1

Maya Carnie, Xiangning He, Olivia Liang, Lauren Moore, Ilana Novakoski, Will Wu;
carni015, he000273, liang625, moor1985, novak560, wuxx1066

February 01, 2020

```
suppressPackageStartupMessages({
  library(TSA)
  library(forecast)
  library(ggplot2)
  library(dplyr)
})
```

```
## Warning: package 'TSA' was built under R version 3.6.2
```

```
## Warning: package 'forecast' was built under R version 3.6.2
```

Binary Random Walk

Assume Y_t is a ***binary*** random walk as we illustrated in the class, such that

$$\begin{cases} Y_1 = e_1 \\ Y_2 = e_1 + e_2 \\ \vdots \\ Y_t = \sum_{i=1}^{t} e_i \end{cases}$$

where $e_i \in \{-1, 1\}$ i.i.d

Note that this is not the Gaussian random walk where $e_t \sim N(0,1)$ but a binary one where e_t can only take two values: -1 and 1 each with probability 50%.

Question 1

a) (1 credit)

Please compute analytically:

$E[Y_3] =$

```
# E(e) = 0
E_Y3 <- 0
```

(please include only the numerical answer above not the computation. An acceptable format for your answer is such as `E_Y3 <- 5`. Please do not rename this variable)

b) (1 credit)

Please compute analytically:

$\mathrm{Var}[Y_3] =$

```r
# Var[e] = 1
Var_Y3 <- 3
```

c) (1 credit)

Please compute analytically:

$\mathrm{Cov}(Y_2, Y_3) =$

```r
Cov_Y2Y3 <- 2
```

d) (1 credit)

Please compute analytically:

$\mathrm{Cov}(Y_5, Y_{10}) =$

```r
Cov_Y5Y10 <-  5
```

Question 2

a) (1 credit)

Please generate one sample path of length $T = 100$ for this binary random walk.

- Please save it into a data.frame `df2a` column `df2a$Y`

```r
set.seed(42) # Please do not change the seed

T <- 100L
e = sample(c(-1,1), size=T, replace=TRUE, prob=c(0.5,0.5))

df2a <- data.frame(Y = cumsum(e), X = time(e))
```

b) (2 credits)

Please generate $N = 100$ sample paths of length $T = 100$ for this binary random walk.

- Please save the results into a data.frame `df2b` where:
  - column `df2b$Y` has the values of the process
  - column `df2b$id` has the id of the sample path
  - column `df2b$t` has the time

```r
set.seed(42) # Please do not change the seed

N <- 100L
T <- 100L

e = sample(c(-1,1), size=N*T, replace=TRUE, prob=c(0.5,0.5))

df2b <- data.frame(Y=e,
                   id=rep(1:N,each=T),
                   t=rep(1:T,N)) %>% as_tibble()
```

c) (1 credit)

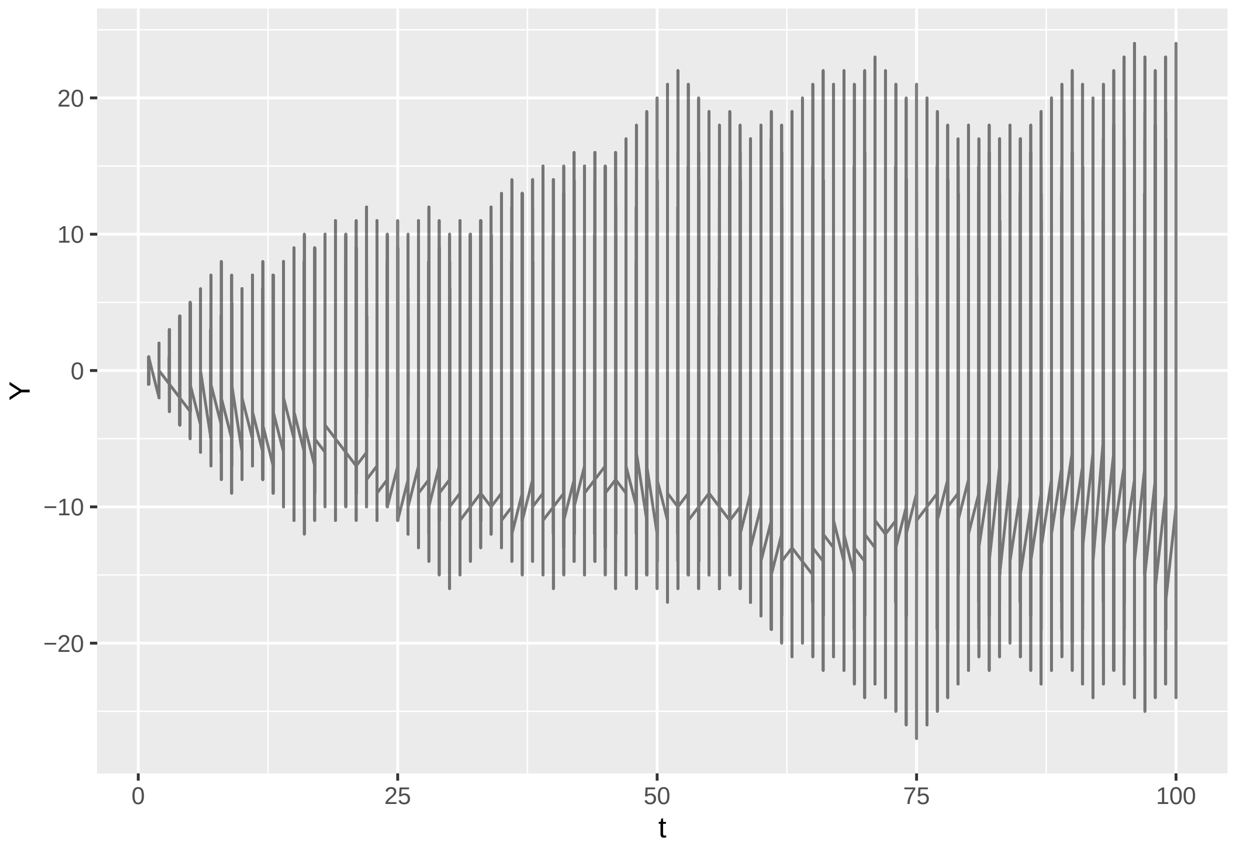
Please plot the sample paths that you generated in the previous question

- Please save your plot into variable `p2c` and plot it

Hints:

- use `ggplot` and take advantage of the long format of the data
- please don't change the color (keep the lines black) but do put `alpha=0.5` into your `geom_line` to make sample paths somewhat transparent.
- do not use `geom_points` just `geom_line` is fine
- As you will see from your plot:
  - the fainter the line the less likely the random walk would reach this spot

```r
df2b1 <- df2b %>% group_by(id) %>% mutate(Y = cumsum(Y))
p2c <- ggplot(data=df2b1,
        aes(x=t, y=Y)) +
        geom_line(alpha=0.5)
p2c
```



d) (1 credit)

Please use the code that you wrote before to generate $N = 5000L$ sample paths.

- Please save the result into `df2d` data.frame following conventions of the prior question
  - column `df2d$Y` has the values of the process
  - column `df2d$id` has the id of the sample path
  - column `df2d$t` has the time

```r
set.seed(42) # Please do not change the seed

N <- 5000L
T <- 100L

e = sample(c(-1,1), size=N*T, replace=TRUE, prob=c(0.5,0.5))

df2d <- data.frame(Y=e,
                   id=rep(1:N,each=T),
                   t=rep(1:T,N)) %>% as_tibble()
df2d <- df2d %>% group_by(id) %>% mutate(Y = cumsum(Y))
```

e) (2 credits)

Use the data in `df2d` to numerically verify your analytical results in Question 1 (a,b,c,d)

```
#Please enter your code here first, AND then answer:
df3 = df2d %>% filter(t==3)

E_Y3 <- mean(df3$Y)
# 0.0068

Var_Y3 <- var(df3$Y)
# 3.013356

df2 = df2d %>% filter(t==2)
Cov_Y2Y3 <- cov(df2$Y, df3$Y)
# 2.005196

df5 = df2d %>% filter(t==5)
df10 = df2d %>% filter(t==10)
Cov_Y5Y10 <-  cov(df5$Y, df10$Y)
# 5.225461
```

Hints:

- For Y_3, extract rows corresponding to `[df2d$t==3]`.
- Since 5000 sample paths are used, the results should be very close to the analytical ones

f) (1 credit)

Assume now that you just got a phone call from your friend telling you that this random walk has just been observed to pass through the point $Y_6 = 6$.

- What that means is you know that all first 6 steps of the random walk happened to be in the "up" direction not "down".

In other words, as this new information from your friend became you can refine your forecast: you can remove the sample paths that the process has not taken and only concentrate on the remaining sample paths that it could still take

- This idea is simulation-based forecasting

Please create data.frame `df2e` such that it only has the sample paths (and only those sample paths) from the previous answer that satisfy the condition that $Y_6 = 6$.

Hints:

- use `group_by`
- the cleanest way is to use dplyr's `do()`
- less clean but also doable: you could create a separate data.frame with acceptable sample path ids and then `inner_join` it to the main data.frame to keep only the good sample paths in it.

```
a <- df2d %>% group_by(id)%>%filter(Y == 6&t == 6)%>%select(id)
df2e <- df2d %>% filter(id %in% a$id)
```

g) (1 credit)

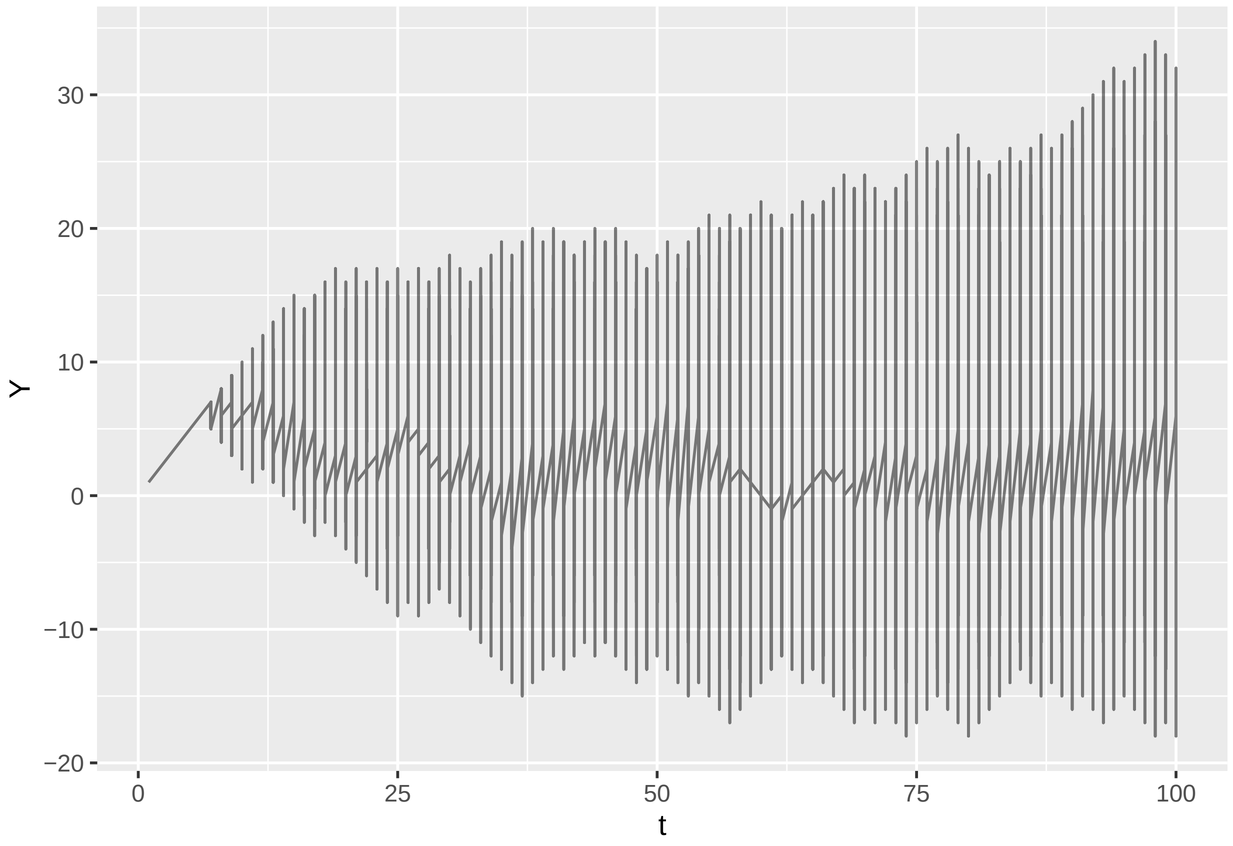
Please use the data.frame from the previous question to plot your results.

- Please don't change the color (keep the lines black) but do put `alpha=0.5` into your `geom_line` to make sample paths somewhat transparent.
- Please do plot the first 6 time periods as well. Yes, the first 6 steps will be exactly the same for all sample paths (since all other sample paths have been eliminated).

Key takeaways:

- You should see from your plot why people say that random walks have memory and are not mean reverting.
- This idea of sequentially eliminating sample paths as you observe the process in order to forecast the future better is called *filtration*. This idea is fundamental in continuous-time stochastic processes but we will only touch upon it for discrete-time stochastic processes.

```r
# please plot it here
p2c <- ggplot(data = df2e, aes(x = t, y = Y))+geom_line(alpha = 0.5)
p2c
```



Problem 2 (11 credits)

HW1

Maya Carnie, Xiangning He, Olivia Liang, Lauren Moore, Ilana Novakoski, William Wu; carni015, he000273, liang625, moor1985, novak560, wuxx1066

February 04, 2020

```
suppressPackageStartupMessages({
  library(TSA)
  library(forecast)
  library(ggplot2)
  library(dplyr)
})
```

```
## Warning: package 'TSA' was built under R version 3.6.2
```

```
## Warning: package 'forecast' was built under R version 3.6.2
```

Moving average with the constant trend

Assume Y_t is a moving average process with the constant trend such as

$$Y_t = (e_t + e_{t-1} + e_{t-2})/3$$

where $e_i \sim N(0, 1)$ i.i.d

Question 1

a) (1 credit)

Please compute analytically the mean:

$E[Y_3] =$

```
E_Y3 <- 0

# phi is <0 and ei is drawn from a normal distb centered about 0
```

(please include only the numerical answer above not the computation. An acceptable format for your answer is such as `E_Y3 <- 5`)

b) (1 credit)

Please compute analytically the variance:

$\text{Var}[Y_3] =$

```r
Var_Y3 <- 1/3

#var = 1/(1 - phi^2)
```

c) (1 credit)

Please compute analytically the autocovariance at lag 1:

$\mathrm{Cov}(Y_5, Y_4) =$

```r
Cov_Y5Y4 <- 2/9

#phi / (1-phi^2)
```

d) (1 credit)

Please compute analytically the autocovariance at lag 2:

$\mathrm{Cov}(Y_6, Y_4) =$

```r
Cov_Y6Y4 <- 1/9
#phi^2 / (1- phi^2)
```

e) (1 credit)

Please compute analytically the autocovariance at lag 3:

$\mathrm{Cov}(Y_7, Y_4) =$

```r
Cov_Y7Y4 <-  0
```

f) (1 credit)

Please compute analytically the autocovariance at lag 4:

$\mathrm{Cov}(Y_8, Y_4) =$

```r
Cov_Y8Y4 <- 0
```

g) (1 credit)

Is the process stationary?

```r
Stationarity <- TRUE #enter TRUE or FALSE
# because phi is < 1
```

Question 2

a) (2 credits)

Please generate $N = 100$ sample paths of length $T = 100$ for this stochastic process.

- Please save the results into a data.frame `df2b` where:
  - column `df2b$Y` has the values of the process
  - column `df2b$id` has the id of the sample path
  - column `df2b$t` has the time

```
set.seed(42) # Please do not change the seed

N <- 100L
T <- 100L

e = rnorm(N*T)


df2b <- data.frame( Y = e, #create time series
  id = rep(1:N, each=T), #repeat time index
  t=rep(1:T,N) #repeat entire vector
) %>% as_tibble()

df2b <- df2b %>% group_by(id) %>% mutate(Y=((Y+zlag(Y)+zlag(zlag(Y)))/3))
```

b) (1 credit)

Please plot the sample paths that you generated in the previous question

- Please save your plot into variable `p2c`

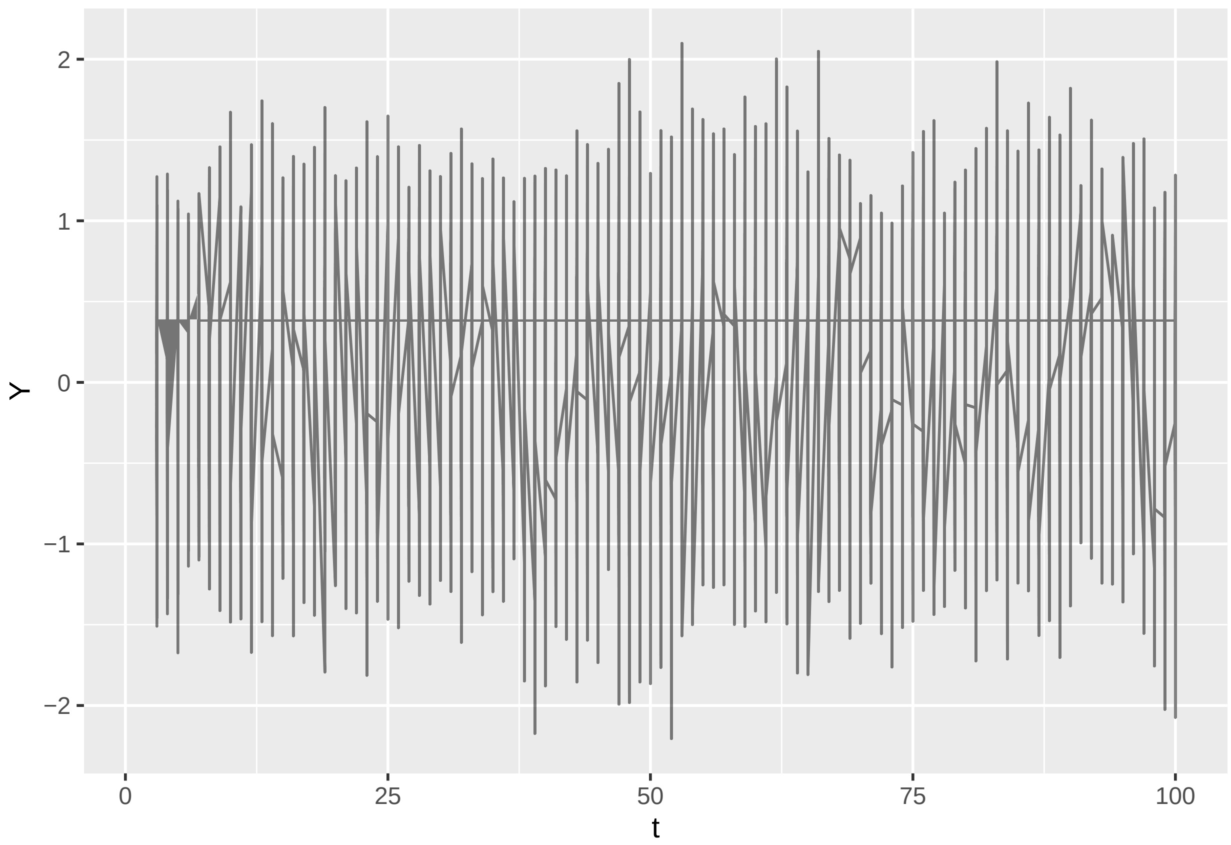
Hints:

- use `ggplot` and take advantage of the long format of the data
- please don't change the color (keep the lines black) but do put `alpha=0.5` into your `geom_line` to make sample paths somewhat transparent.
- do not use `geom_points` just `geom_line` is fine
- As you will see from your plot:
  - the fainter the line the less likely the stochastic process would reach this spot

```
#p2c <- ggplot(<here you go>)
#p2c

p2c = ggplot(data = df2b,aes(x=t, y=Y)) + geom_line(alpha = 0.5)
p2c
```

```
## Warning: Removed 200 rows containing missing values (geom_path).
```



- You should be able to clearly see that this process is indeed stationary.
    - The density of black color represents the probabilty of finding the process in this place.

c) (1 credits)

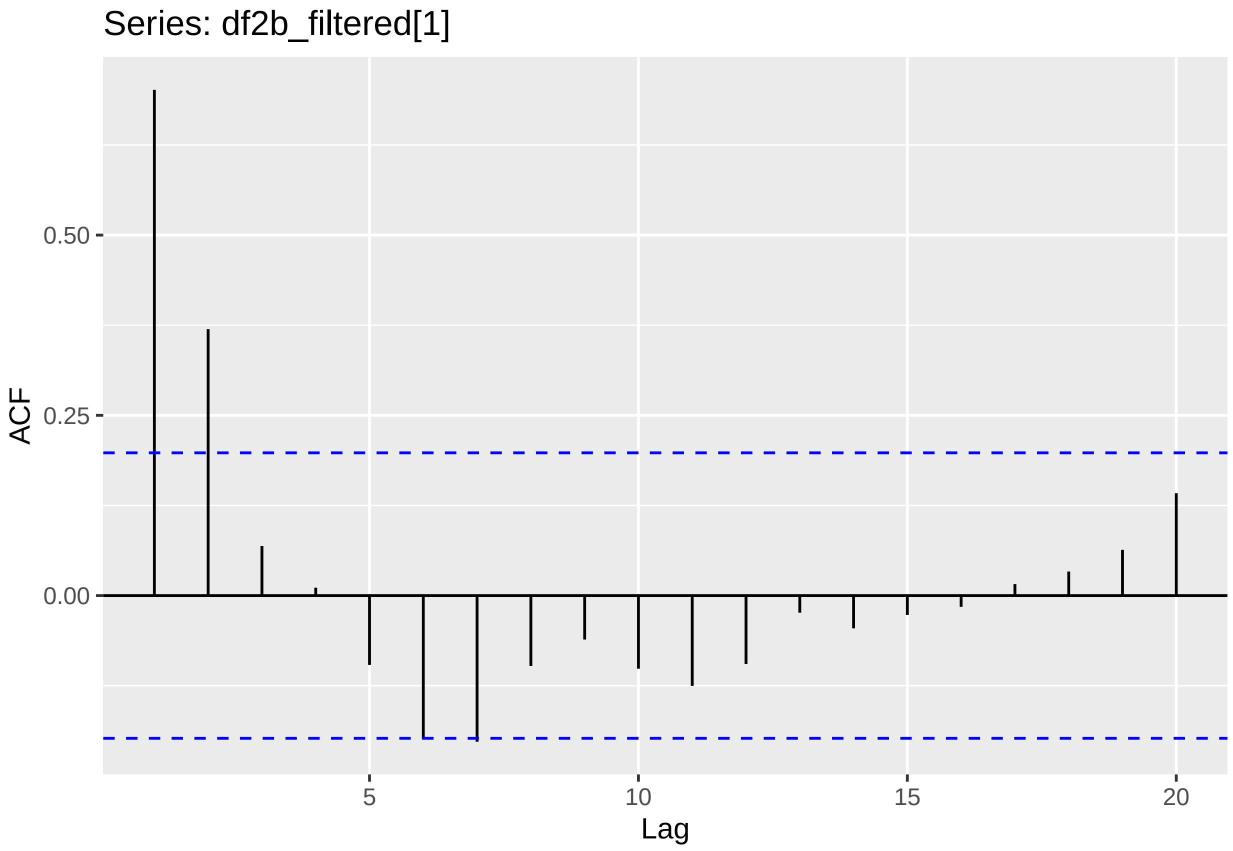
Please pick a sample path with `id=1` from `df2b` and plot *an autocorrelation function (ACF)*.

- Autocorrelation function plot is a plot of approximately 20 different correlations estimated from one sample path. For example, point 1 on X-axis of that plot corresponds to your estimate of $\mathrm{Corr}(X_t, X_{t-1})$ (this is called *autocorrelation at lag 1*), point 2 on X-axis of that plot corresponds to your estimate of $\mathrm{Corr}(X_t, X_{t-2})$ (*autocorrelation at lag 2*) and so on.
- Autocorrelation plots also contain the (dotted blue) lines that corresponds to the noise threshold. All autocorrelations within the band are indistinguishable from noise (statistically insignificant). However, autocorrelations at some lags will clearly stick outside of that band.
- The work that you did in class over the previous sessions should start paying off now. This autocorrelation plot will clearly reveal something interesting: you have a strong auto-correlation at lags 1 and 2 and nothing beyond that. This is a clear indication of the moving average process of an order 2.

Hints:

- use `ggAcf` function from forecast package

```
df2b_filtered = df2b %>% filter(id==1)
ggAcf(df2b_filtered[1])
```



- The analytical work that you did in class over the previous sessions should start paying off now. Please look back all the way to problem 2 Q1 and compare to your answers to Q1 e) to Q1 f). Autocorrelation is standardized autocovariance. You will find that a zero autocovariance leads to a very small autocorrelation in the ACF.
- This autocorrelation plot will clearly reveal something interesting: you do indeed have a strong autocorrelation at lags 1 and 2 and almost zero beyond that. In other words, when you work with real data and you happen to get this ACF with two peaks at lag 1 and lag 2 – you should immediately understand what type of process tends to generate such ACF plot – moving average!
- (no need to report anything about your conclusions here)

Problem 3 (6 credits)

HW1

Maya Carnie, Xiangning He, Olivia Liang, Lauren Moore, Ilana Novakoski, William Wu; carni015, he000273, liang625, moor1985, novak560, wuxx1066

February 04, 2020

```
suppressPackageStartupMessages({
  library(TSA)
  library(forecast)
  library(ggplot2)
  library(dplyr)
})
```

```
## Warning: package 'TSA' was built under R version 3.6.2

## Warning: package 'forecast' was built under R version 3.6.2
```

Boston Crime Data

Crime incident reports are provided by Boston Police Department (BPD) to document the initial details surrounding an incident to which BPD officers respond. This data set contains the date of all crimes from 6/15/2015 to 9/3/2018. We are interested in knowing the frequency of crimes changed over months.

Question 1 (1 credit)

Please change your working directory and load the data `crime.txt`. Report the dimension of the data.

Hints:

- Set `header=T`

```
#Please insert your code below
crime=read.delim('crime.txt', header=T, sep = '')
```

Question 2 (1 credit)

Please aggregate the data based on their date. That is, we should end up with a smaller dataset where each row contains year, month, day, and the frequency of crimes on that date. Report the dimension of the new dataset.

Hints:

- Create an all-one vector having the same length as the data, then consider the `aggregate` function where you could set the `list` option for grouping elements, and set the `FUN` option as `sum`.
- Aggregating data is an important skill for almost everyday data cleaning.

```
#Please insert your code below
crime_ts <- crime %>% group_by(Year, Month, Day) %>% summarise(cnt=n())

#aggregate(crime, by = list(crime$Year, crime$Month, crime$Day), FUN = length)
```

Question 3 (1 credit)

Sort the data by Year, Month, and Day. Report the first ten rows of the sorted data

Hints:

- Consider the `order` function

```
#Please insert your code below
crime_ts=crime_ts[order(crime_ts$Year, crime_ts$Month, crime_ts$Day), ]

head(crime_ts,10)
```

```
## # A tibble: 10 x 4
## # Groups:   Year, Month [1]
##     Year Month   Day   cnt
##    <int> <int> <int> <int>
##  1  2015     6    15   249
##  2  2015     6    16   249
##  3  2015     6    17   234
##  4  2015     6    18   294
##  5  2015     6    19   289
##  6  2015     6    20   261
##  7  2015     6    21   186
##  8  2015     6    22   262
##  9  2015     6    23   266
## 10  2015     6    24   276
```

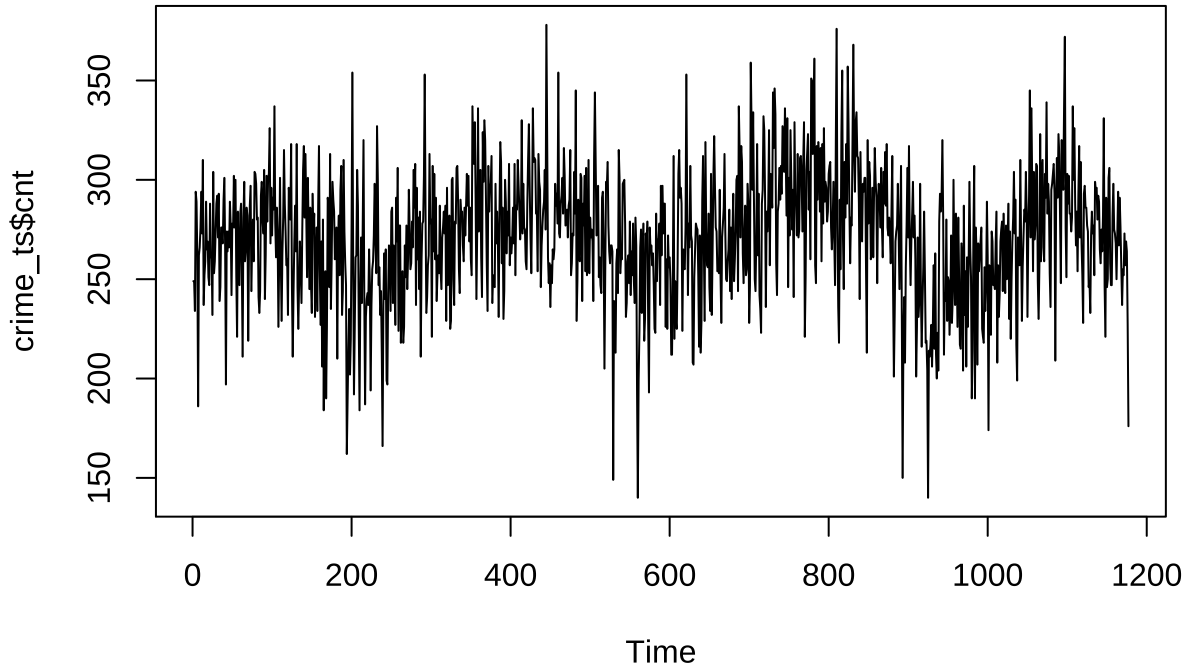
Question 4 (1 credit)

Plot the frequency of crimes by date. Do you see a pattern?

Hints:

- Consider the `ts.plot` function

```
#Please insert your code below
ts.plot(crime_ts$cnt)
```



Question 5 (1 credit)

Is the time series stationary? Why?

Hints:

- Recall the definition of stationarity. What's the requirement on the mean function?

```
#Please insert your answer below

#It is likely to be stationary by viewing the gragh.
#The data fluctuates around a fixed mean with relatively stable correlation and covariance over time.
E_cnt <- mean(crime_ts$cnt)
Var_cnt <- var(crime_ts$cnt)
Cov_cnt <- cov(crime_ts$cnt[1:1176],crime_ts$cnt[2:1177])
# By calculating the mean, correlation and covariance, we found that they are all constant.
fixed = 1175/5
E_1 = mean(crime_ts$cnt[1:fixed])
E_2 = mean(crime_ts$cnt[fixed +1:fixed*2])
E_3 = mean(crime_ts$cnt[fixed*2 +1:fixed*3])
E_4 = mean(crime_ts$cnt[fixed*3 +1:fixed*4])
E_5 = mean(crime_ts$cnt[fixed*4 +1:1177])
# Further checking the mean of several parts of the time series using random cutoff,
# we get very similary means for different parts of the time series.
```

```
#Conclusion: from the observations above, we are confident enough that this time series is stationary.
```

Question 6 (1 credit)

Which date has the highest crime frequency? How many crimes were reported on that day?

```
#Please insert your code and answer below
crime_ts[crime_ts$cnt == max(crime_ts$cnt),]
```

```
## # A tibble: 1 x 4
## # Groups:   Year, Month [1]
##    Year Month   Day   cnt
##   <int> <int> <int> <int>
## 1  2016     9     1   378
```