Problem 1 (9 credits)

HW2

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

February 11, 2020

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

White noise

General Requirements

- Please do not change the path in `readRDS()`, your solutions will be automatically run by the bot and the bot will not have the folders that you have.
- Please review the resulting PDF and make sure that all code fits into the page. If you have lines of code that run outside of the page limits we will deduct points for incorrect formatting as it makes it unnecessarily hard to grade.
- Please avoid using esoteric R packages. We have already discovered some that generate arima models incorrectly. Stick to tried and true packages: base R, `forecast`, `TSA`, `zoo`, `xts`.

Problem Description

This problem is inspired by my previous colleague's first encounter with interesting characteristics of white noise back at Samsung Electronics more than a decade ago.

A fellow engineer was working on GPS navigation devices and what was really curious to him was that the *Signal-to-Noise Ratio (SNR)* for GPS by design is *negative* meaning that the ambient radio noise is stronger than signal, in fact way stronger! Yet the device works!

As a human looking at this type of data, it is impossible to spot any patterns in it – the time series looks like a white noise and any useful signal is too faint to be seen. However, with the clever use of math, the engineers are able to recover this faint signal from the remote satellites despite the fact that it is being completely overpowered by terrestrial noise sources.

For this problem, we will look at one version of this problem in the time domain[1]. The key observations that you need to use here:

- the ambient radio noise is white noise
- the satellite sends the same (or similar) thing over and over again.

[1] Please note that this formulation is not exactly how GPS receiver works but rather a simplified problem that is inspired by it

Given that, you can theoretically recover any signal no matter how faint from any levels of noise by repeating it enough times. In other words, high levels of noise impact the speed of data transfer rather than the possibility.

For this problem, please load the noisy signal data from file `problem1.Rds`

```
problem1 <- readRDS("problem1.Rds") # Please do not change this line
```

This file contains a (simulated) noisy signal. Note that the data is generated such that:

- Var(Signal) = 1
- Var(Noise) = 100

$$\mathrm{SNR} = 10 \cdot \log\left(\frac{\mathrm{Var(Signal)}}{\mathrm{Var(Noise)}}\right) \text{ decibel}$$

In other words, the noise is 100x more powerful than the signal and SNR is negative -20 decibel. Don't try to spot the signal in the raw time series - you will not be able to.

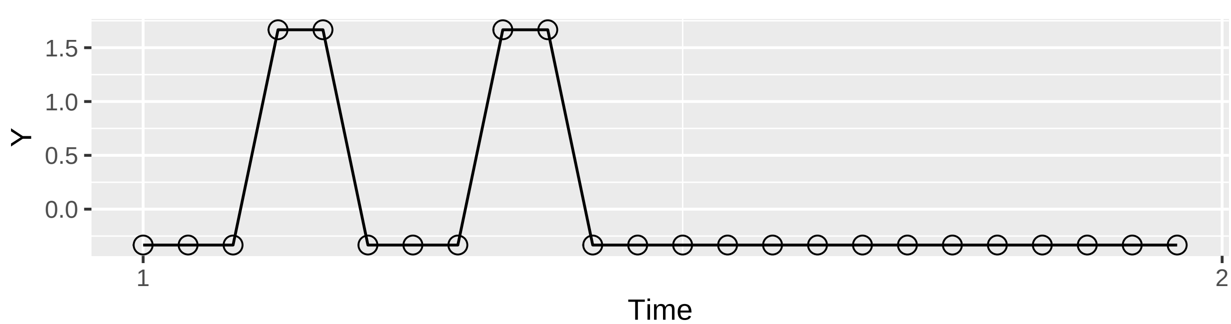
The signal's seasonality period is 24 (which means that the true signal is repeated by the satellite every 24 observations).

The true signal that is sent by the satellite is a sequence of pulses that look somewhat like a plot below. These pulses can be used to represent 0s and 1s. As an example, the plot contains 2 pulses.

```
signal <- c(0,0,0,2,2,0,0,0,2,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0)  # 24 signals
signal <- signal - mean(signal)

Y <- ts( signal, frequency = length(signal))

autoplot(Y) + geom_point(size=3, shape=1)
```



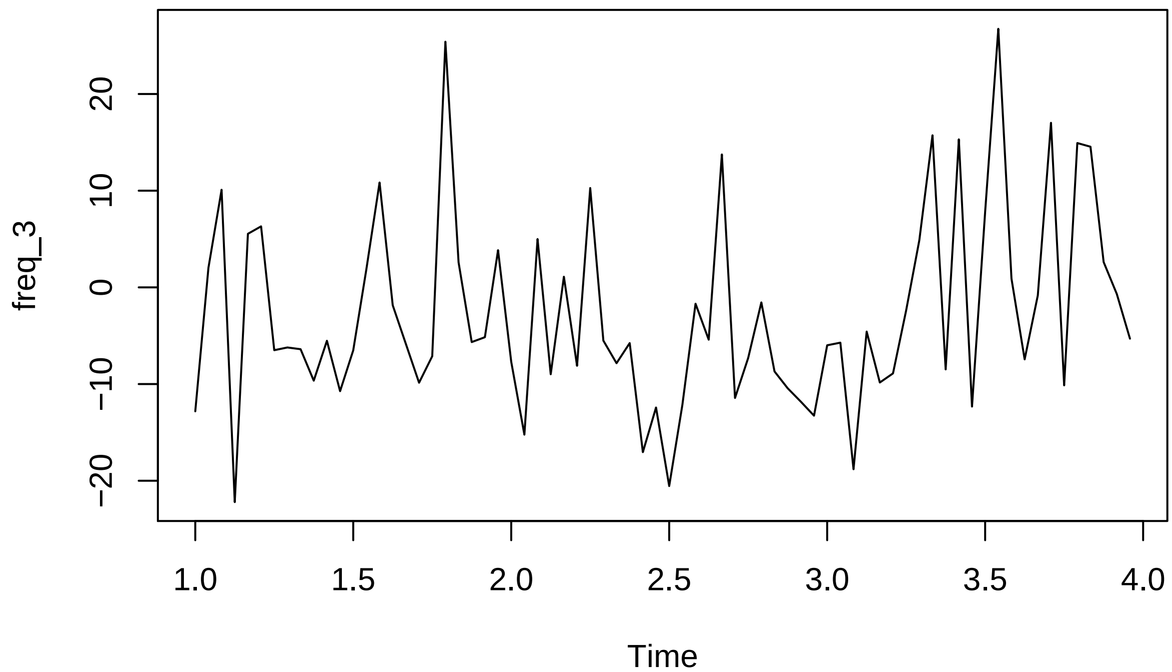
Question 1 (2 credits)

Please read the data and plot the first 72 observations from the noisy signal data in `problem1`.

The true signal has been repeated 3 times by that time but you can't see it – it is completely overwhelmed by the background noise (the signal's scale is around ±1, while the noise scale is around ±10)

```
# Plot here

freq_3 <- ts(data.frame(problem1),start=c(1, 1),end=c(3, 24), frequency = 24)
ts.plot(freq_3)
```



Q2 (3 credits)

Please figure out how to remove all the (Gaussian) white noise and plot the signal (sequence of pulses) that you recovered. The true signal has been repeated so many times in `problem1` that it should be very easy to recover it.

For Q2, please do it by averaging "manually" and without using any time decomposition functions from the forecast package like `decompose`, `ma` or `stl`.

Note:

- your plot may not look as clean as the sample signal that I plotted above but the pulses will be very clear and visible nevertheless.

Hint:

- Use `cycle(x)` function to recover the season for each value
- In base R, you can use `apply` or `aggregate` function with a formula to compute the means.
- In `dplyr`, you can use `group_by` and `summarise`.

Output:

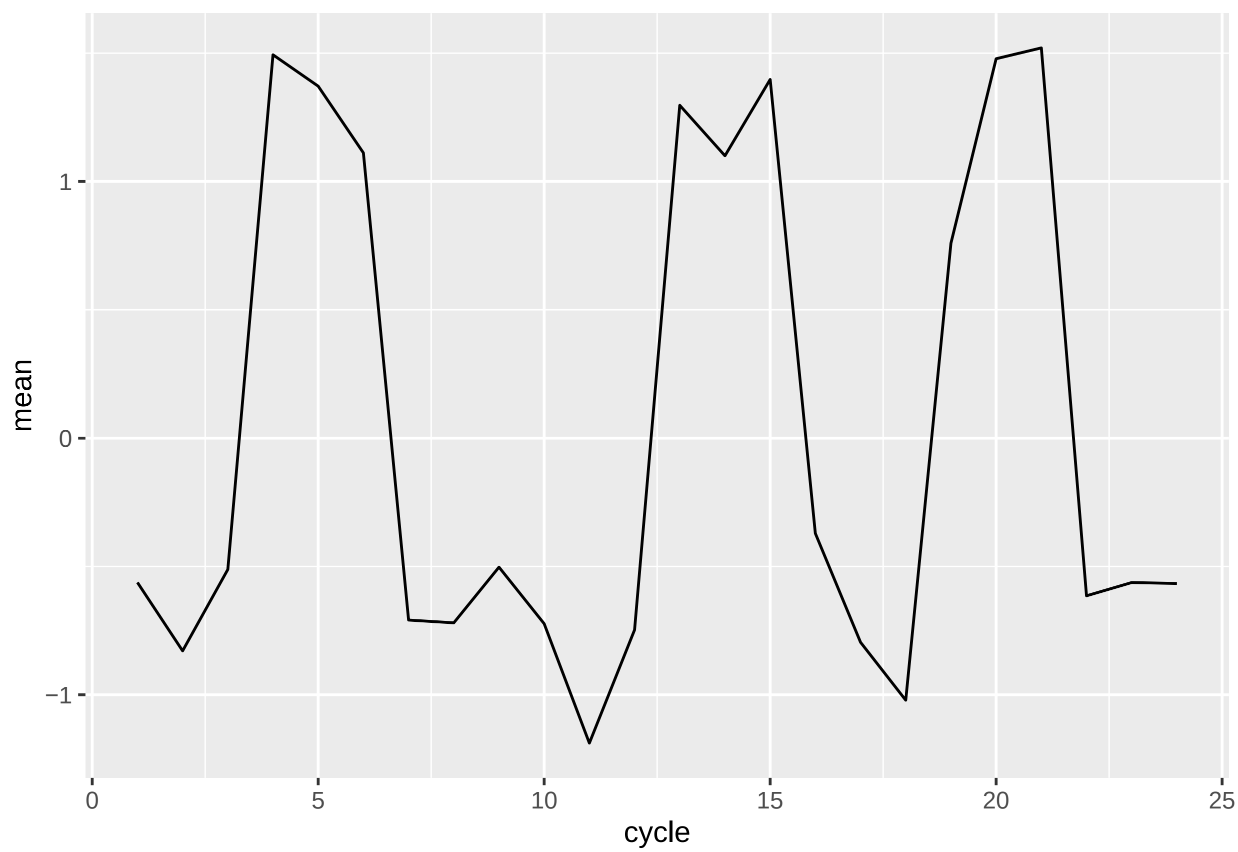
- please produce a vector `q2_means` of length 24 that contains the recovered signal (in other words, the seasonal means)
- please plot your result

```
# Please write your code here

q2_df <- data.frame("signal" = problem1,
                    "wn" = rnorm(120000, sd =10),
                    "cycle" = cycle(problem1))

q2_df <- q2_df %>% mutate('true_signal' = signal - wn)
mean_df <- q2_df %>% group_by(cycle) %>% summarize(mean = mean(true_signal))
#q2_means <- rep(NA, frequency(problem1))
q2_means <- mean_df$mean

ggplot(mean_df , aes(y = mean, x = cycle)) + geom_line()
```



Q3 (2 credits)

As you recovered the true signal, how many pulses are there in one time window of length 24 observations?

Output:

- please a numeric value `q3_num_pulses` that contains the number of pulses that you saw on the plot.

```
# Please write down your answer here:


q3_num_pulses <- 3
```

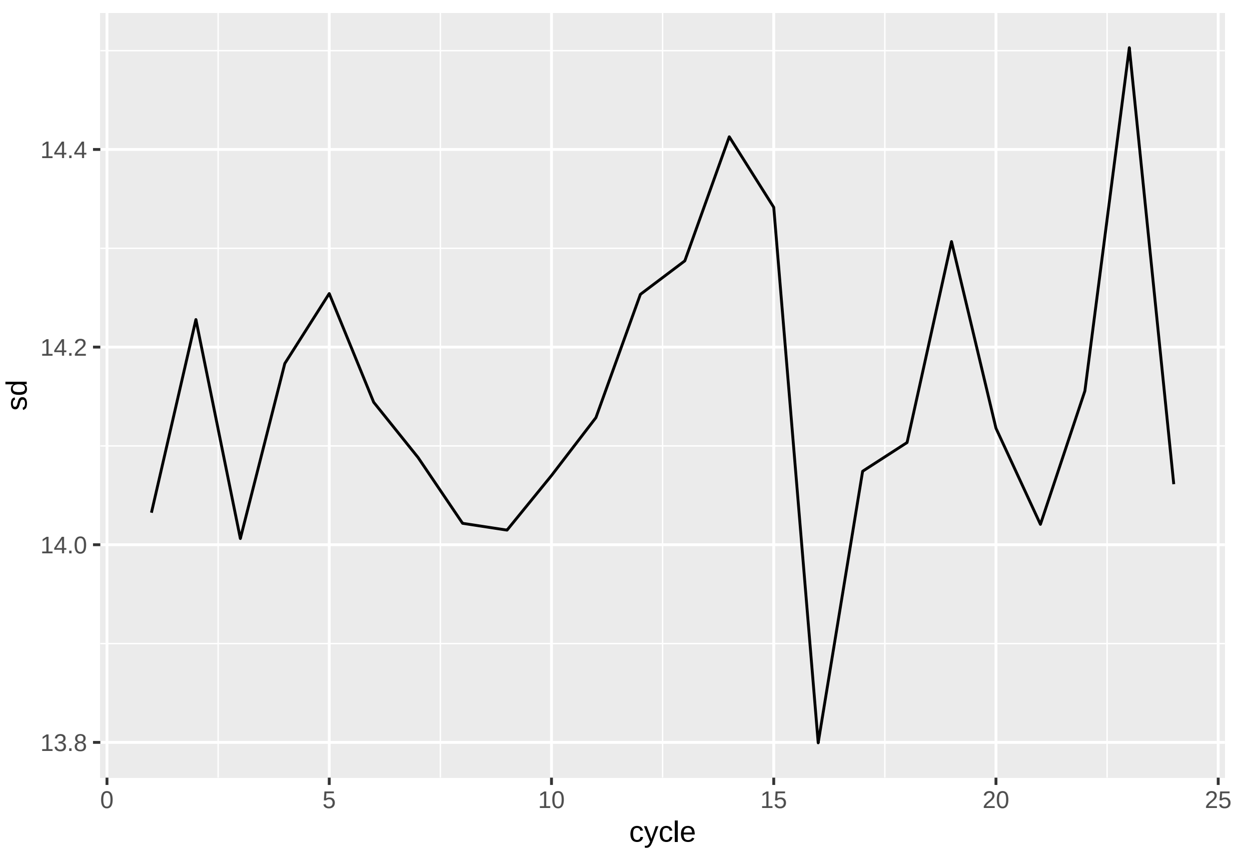
Q4 (2 credits)

Please produce a vector `q4_sd` of length 24 that contains the standard deviation of the recovered signal Display or plot the 24 standard deviations.

```
# Please write your code here

sd_df <- q2_df %>% group_by(cycle) %>% summarize(sd = sd(true_signal))
#q4_sd <- rep(NA, frequency(problem1))
q4_sd <- sd_df$sd

ggplot(sd_df , aes(y = sd, x = cycle)) + geom_line()
```



Note:

- you can see how large the standard deviations are, compared with the means in Question 2. Yet the signals are still identified, due to the large sample.

- please note that `forecast` package includes automated functions that would do time series decomposition for you such as `decompose`, `ma` or `stl`. For Q3, you shouldn't use them.

Take aways:

- Don't be afraid of white noise. In the world of randomness, white noise is a friend not an enemy.
- Be afraid of non-white noise. For GPS engineers, it is not the strong ambient radio noise that is the main issue, it is the faint correlated one – all these faint little reflections of the GPS signal from the nearby skyscrapers and buildings (called "multipath") — this noise is autocorrelated with the original signal and thus, cannot be cancelled out so easily. That's why your GPS tends to misbehave when you are in the middle of a downtown surrounded by the radio-reflective metal skyscrapers.

Problem 2 (10 credits)

HW2

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

February 12, 2020

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

Characteristic Polynomials

Question 1

Assume Y_t is the following stochastic process such as

$$Y_t = 2.2 \cdot Y_{t-1} - 1.57 \cdot Y_{t-2} + 0.36 \cdot Y_{t-3} + e_t$$

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

a) (1 credit)

First, let's determine whether the process is stationary or not by computing the roots of the characteristic polynomial.

Hints:

- use `polyroot()` function

Please pick the smallest root as x_1 and the larger root as x_3:

```
polyroot(c(1, -2.2, 1.57, -0.36))
```

```
## [1] 1.111111-0i 1.250000+0i 2.000000-0i
```

```
x_1 <- 1.11
x_2 <- 1.25
x_3 <- 2
```

(please include only the numerical answer above not the computation. An acceptable format for your answer is such as `x_1 <- 5`)

b) (1 credit)

Based on your answer above conclude whether the process is stationary or not:

```
stationary <- TRUE # type a boolean: TRUE or FALSE

## because all the roots are outside of the scope of (-1,1)
```

c) (2 credits)

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

- Please save the results into a data.frame `df2c` where:
  - column `df2c$Y` has the values of the process
  - column `df2c$id` has the id of the sample path
  - column `df2c$t` has the time

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

N <- 100L
T <- 100L

df2c <- data.frame(Y=rnorm(N*T), id=rep(1:N, each = T), t=rep(1:T,N)) %>% as_tibble()

df2c <- df2c %>% group_by(id) %>% mutate(Y = (Y + arima.sim(model=list(order = c(3,0,0),ar=c(2.2,-1.57,
```

d) (1 credit)

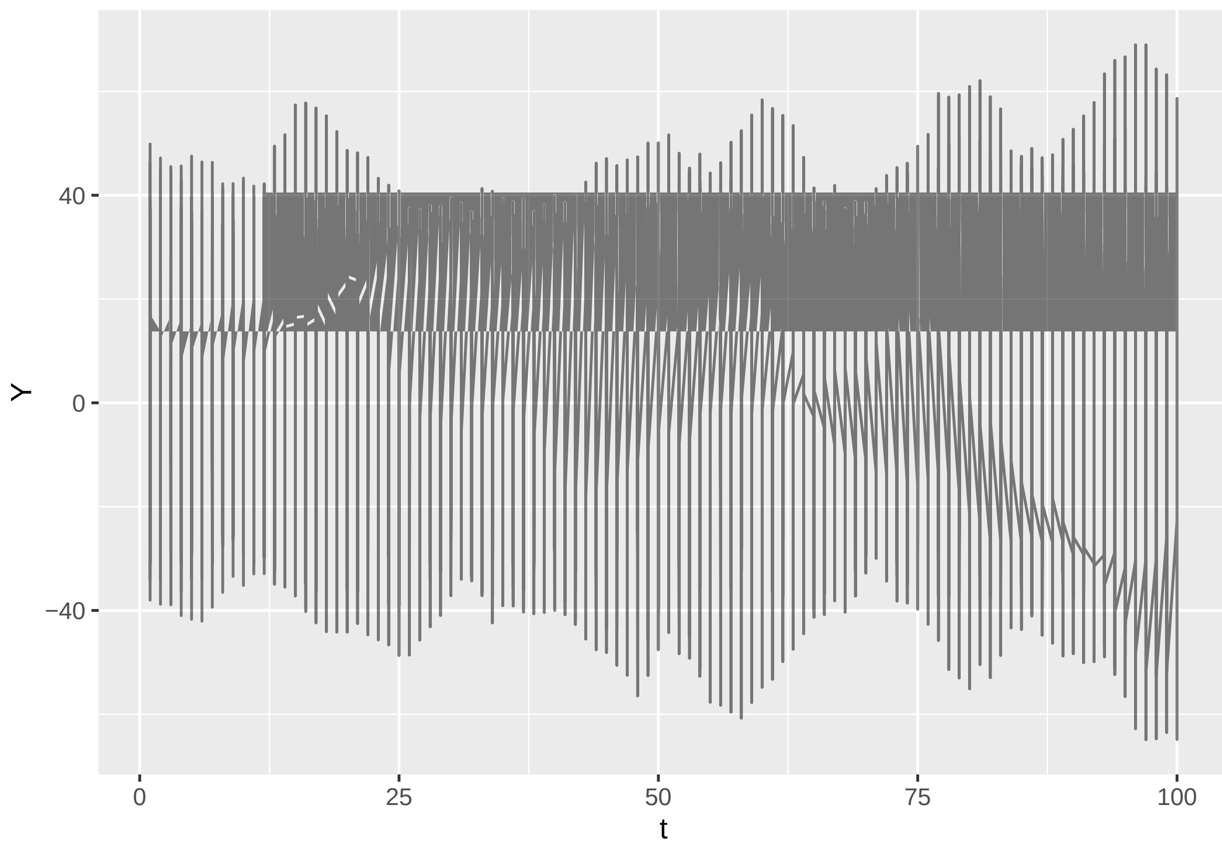
Please plot the sample paths that you generated in the previous question

- Please save your plot into variable `p1d`

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=0.05` 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

```
# ggplot here
p1d = ggplot(data = df2c,aes(x=t, y=Y)) + geom_line(alpha = 0.5)
p1d
```



Question 2 (5 credits)

Repeat a) - d) in Question 1 for the following stochastic process Y_t:

$$Y_t = 2.4 \cdot Y_{t-1} - 1.55 \cdot Y_{t-2} + 0.3 \cdot Y_{t-3} + e_t$$

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

Compared with Question 1, we expect to see significant difference in the stationarity from the plot, although the coefficients are very close.

a)

First, let's determine whether the process is stationary or not by computing the roots of the characteristic polynomial.

Hints:

- use `polyroot()` function

Please pick the smallest root as x_1 and the larger root as x_3:

```
polyroot(c(1, -2.4, 1.55, -0.3))
```

```
## [1] 0.6666667+0i 2.0000000-0i 2.5000000+0i
```

```
x_1 <- 0.67
x_2 <- 2
x_3 <- 2.5
```

(please include only the numerical answer above not the computation. An acceptable format for your answer is such as `x_1 <- 5`)

b)

Based on your answer above conclude whether the process is stationary or not:

```
stationary <- FALSE # type a boolean: TRUE or FALSE
## because there is at lease one root that is within the scope of (-1,1)
```

c)

Please generate $N = 100$ sample paths of length $T = 20$ for this stochastic process.

- Please save the results into a data.frame `df2c` where:
  - column `df2c$Y` has the values of the process
  - column `df2c$id` has the id of the sample path
  - column `df2c$t` has the time

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

N <- 100L
T <- 20L

xx <- vector("numeric",T)
# innovations (process errors)
ww <- rnorm(T)
# set first 3 times to innovations
xx[1:3] <- ww[1:3]

# simulate AR(3)
for(t in 4:(T)) {
  xx[t] <- 2.4*xx[t-1] -1.55*xx[t-2]+0.3*xx[t-3] + ww[t]
}


df2 <- data.frame('Y'= xx)

for (i in 2:N){
  xx <- vector("numeric",T)
# innovations (process errors)
  ww <- rnorm(T)
# set first 3 times to innovations
  xx[1:3] <- ww[1:3]

# simulate AR(3)
  for(t in 4:(T)) {
```

```
    xx[t] <- 2.4*xx[t-1] -1.55*xx[t-2]+0.3*xx[t-3] + ww[t]
  }
  a = data.frame('Y'= xx)
  df2 = rbind(df2,a)
}

df2c <- data.frame(Y = df2$Y, id = rep(1:N, each = T), t = rep(1:T, N))
```

d)

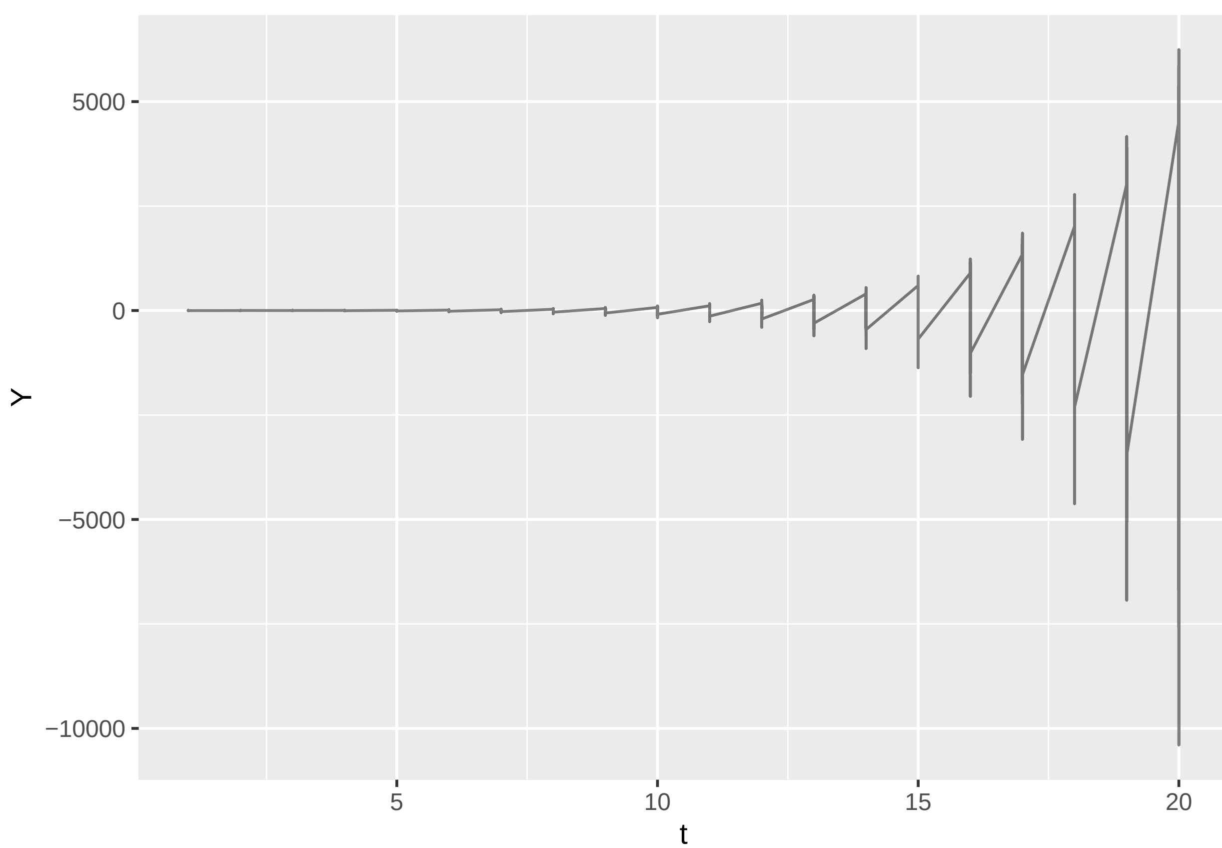
Please plot the sample paths that you generated in the previous question. You should see the effect of the roots of the polynomial on the sample paths of the process.

- Please save your plot into variable `p1d`

Hints:

- use `ggplot` and take advantage of the long format of the data
- do not use `geom_points` just `geom_line` is fine

```
# ggplot here
p1d = ggplot(data = df2c,aes(x=t, y=Y)) + geom_line(alpha = 0.5)
p1d
```



Problem 3 (6 credits)

HW2

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

February 12, 2020

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

Boston Crime Data Analysis

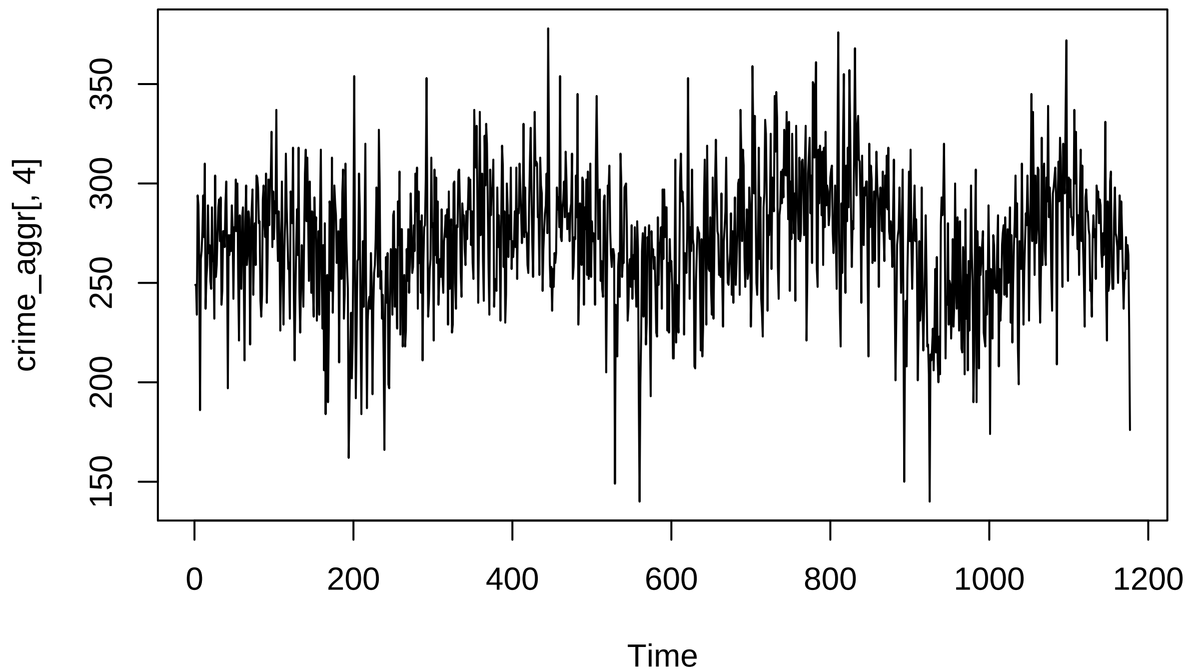
Question 1

Please pull out the crime frequency data we got from Homework 1 - Problem 3 - Question 4. You may re-plot the time series to refresh yourself about the pattern.

```r
crime=read.table("crime.txt",header=T)

N=dim(crime)[1]
crime_aggr=aggregate(rep(1,N),list(year=crime[,1],month=crime[,2],day=crime[,3]),sum)
crime_aggr=crime_aggr[order(crime_aggr[,1],crime_aggr[,2],crime_aggr[,3]),]

ts.plot(crime_aggr[,4])
```



a) (1 credit)

First, let's fit an `auto.arima()` to find out a good ARIMA model for the data. Again, notice that, `auto.arima()` provides a "good" model but not necessarily the optimal. We will learn more concrete model selection techniques in Lecture 6.

Hints:

- use `auto.arima()` function

```
#Please provide your code below
auto.arima(crime_aggr$x)
```

```
## Series: crime_aggr$x
## ARIMA(1,0,3) with non-zero mean
##
## Coefficients:
##          ar1      ma1      ma2     ma3      mean
##       0.9888  -0.7142  -0.2542  0.0446  270.6409
## s.e.  0.0054   0.0298   0.0347  0.0292    5.5130
##
## sigma^2 estimated as 880.5:  log likelihood=-5658.27
## AIC=11328.53   AICc=11328.61   BIC=11358.96
```

b) (2 credits)

What's the model? For example

$$(Y_t - 10) = 0.4 \cdot (Y_{t-1} - 10) + e_t - 0.8 \cdot e_{t-1}$$

Hints:

- The mean value comes with every Y_t. In the example above, the mean value is 10.
- R assumes positive sign for MA models. In the example above, R would show -0.8, rather than +0.8 for the MA(1) coefficient

Please write down the model below:

$$(Y_t - 270) = 0.99 \cdot (Y_{t-1} - 270) + e_t - 0.71 \cdot e_{t-1} - 0.25 \cdot e_{t-2} + 0.04 \cdot e_{t-3}$$

c) (1 credit)

Are any of the coefficients significant?

Hints:

- A coefficient is significant if its magnitude is (roughly) at least twice as large as its standard error.

```r
#Please write down your answer below
## The coefficients of ar1, ma1, and ma2 are all significant,
## while the coefficient of ma3 is slightly not significant.
```

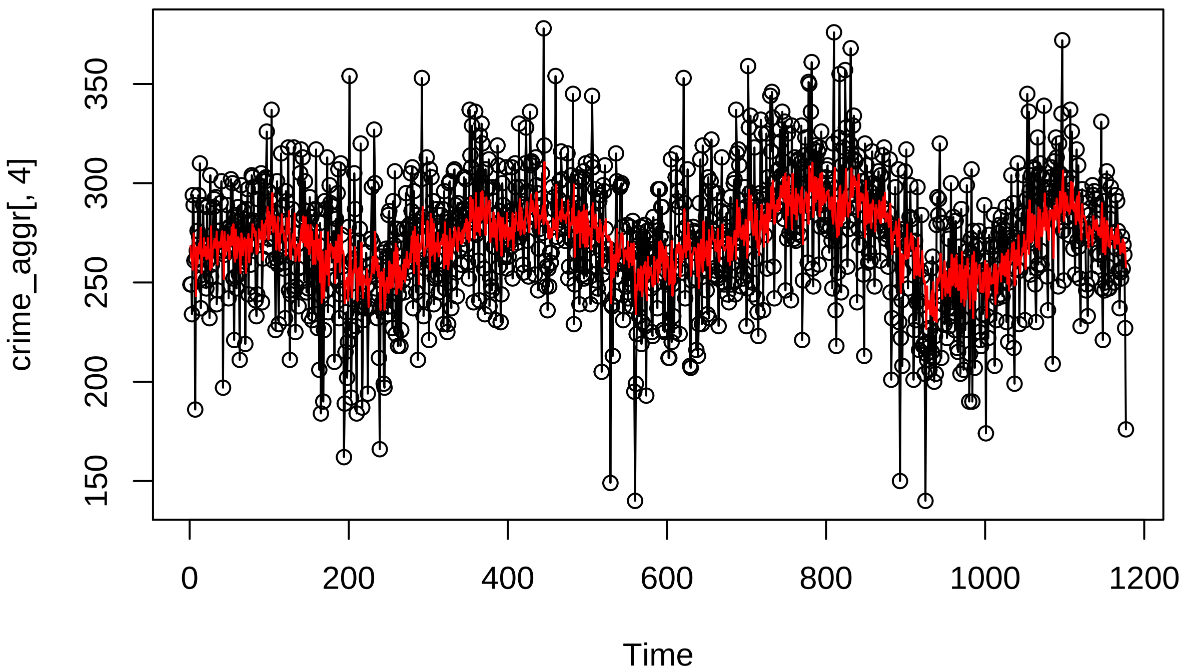
d) (2 credits)

Please superimpose the fitted values on the original crime frequency time series. Does the model sufficiently explain the data?

Hints:

- The fitted values can be calculated by `the original time series - arima_fit$residuals`

```r
# Plot here
arima_fit <- Arima(crime_aggr$x, order = c(1,0,3))
ts.plot(crime_aggr[,4], type = 'o')
lines(arima_fit$fitted, col = 'red')
```




```
## The model does not sufficiently explain the data,
## because it only captures the major trend of the time series but fail to explain the big variance.
```