Problem 1 (9 credits)

HW2

Michael Brucek (mbrucek), Eduardo Chavez (echavezh), Kevin Grady (grady133), Danny Moncada (monca016), Tian Zhang (zhan7003)

March 02, 2020

```
suppressWarnings(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.

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`

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

```r
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

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

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.

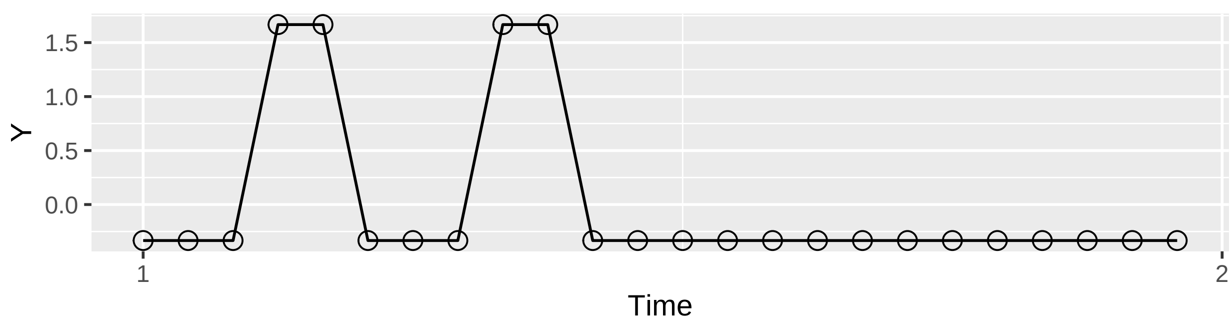
The signal's seasonality period is 24 (which means that the true signal is repeated by the satellite every 24 observations).

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.

```r
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)
signal <- signal - mean(signal)

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

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



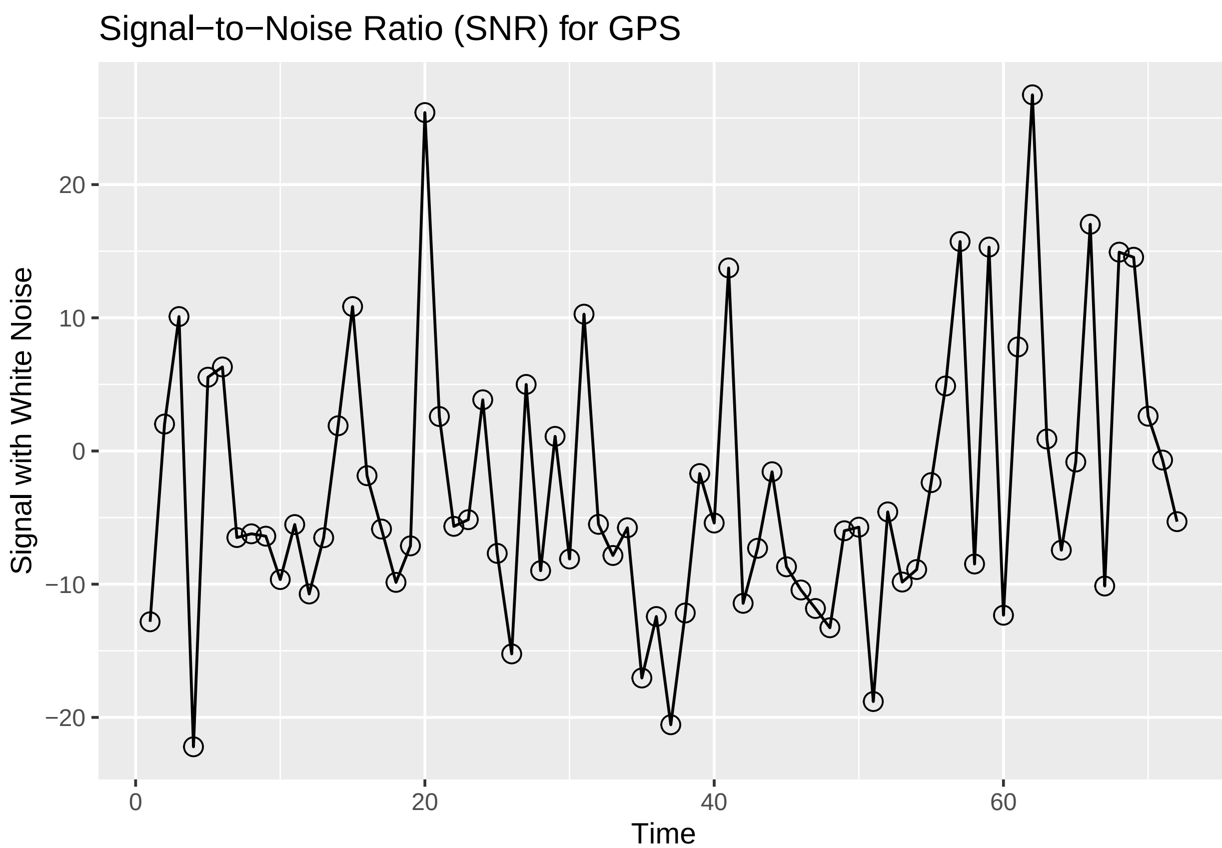
Question 1 (2 credits)

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)

```r
Y = ts(problem1[1:72])

autoplot(Y) + geom_point(size = 3, shape = 1) +
            labs(x = "Time", y = "Signal with White Noise",
                      title = "Signal-to-Noise Ratio (SNR) for GPS")
```



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`.

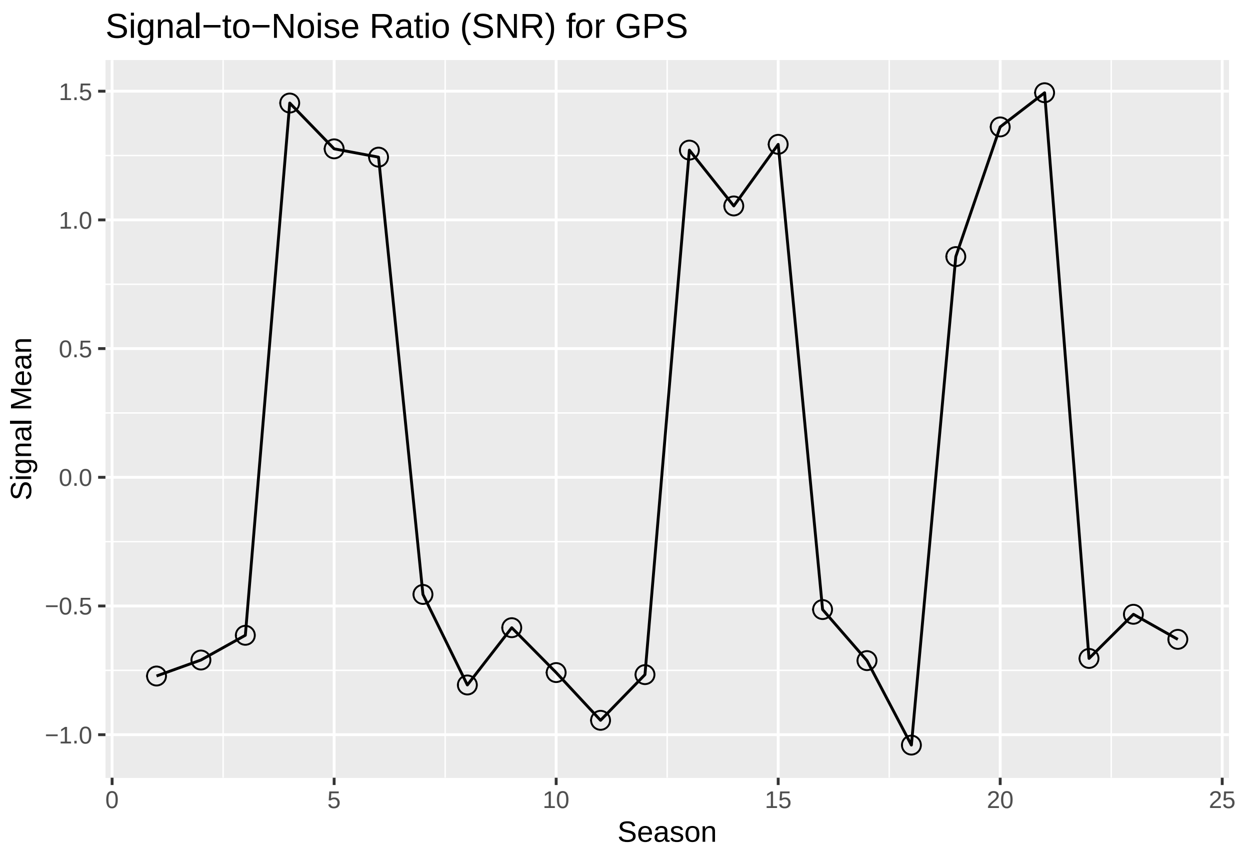
Output:

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

```
## Convert into dataframe for dpylr manipulation
q2_means <- data.frame(seasonal_point = cycle(problem1), signal = problem1) %>%
                    group_by(seasonal_point) %>% summarise(means = mean(signal))
```

```
Y_mu = ts(q2_means$means)

## Plot
autoplot(Y_mu) + geom_point(size = 3, shape = 1) +
        labs(x = "Season", y = "Signal Mean",
                     title = "Signal-to-Noise Ratio (SNR) for GPS")
```



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

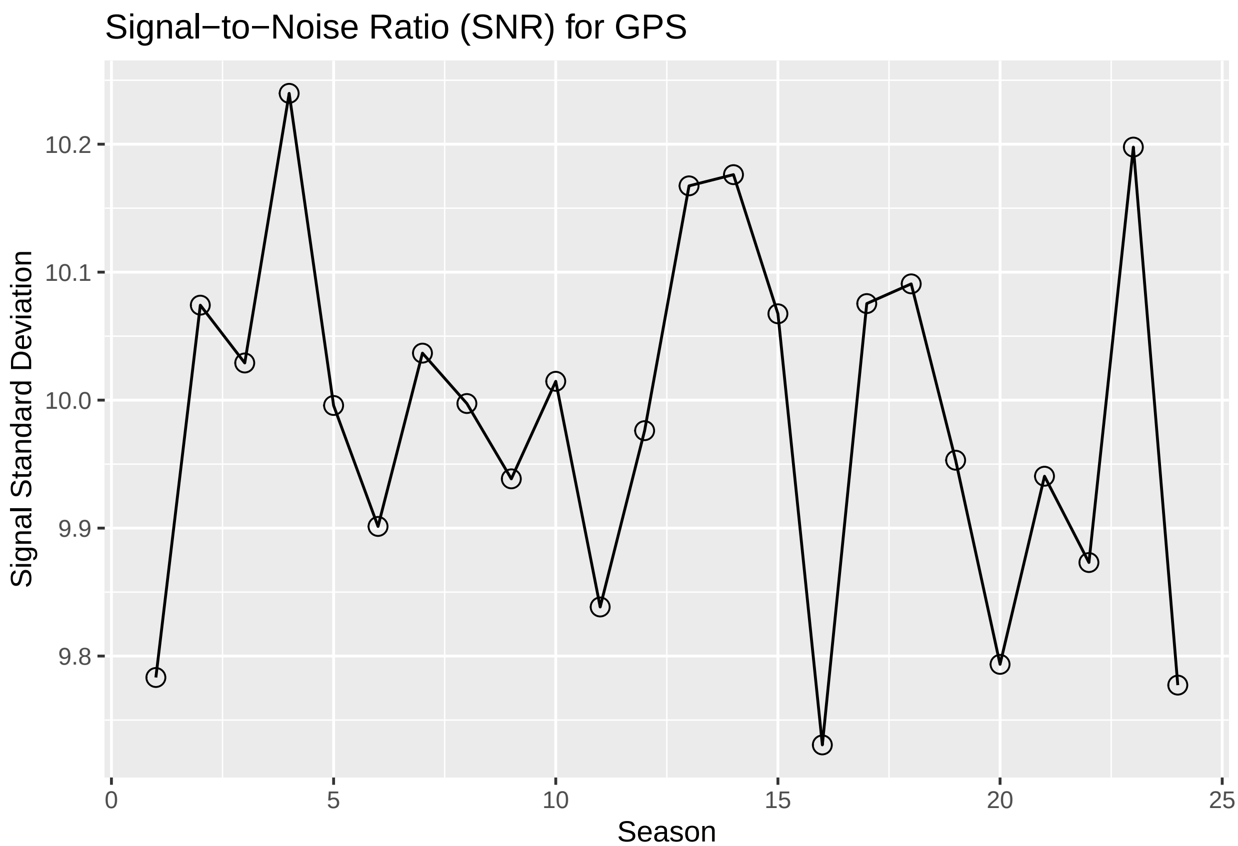
Display or plot the 24 standard deviations.

```
## Convert into dataframe for dpylr manipulation
q4_sd <- data.frame(seasonal_point = cycle(problem1), signal = problem1) %>%
```

```r
          group_by(seasonal_point) %>%
            summarise(standard_dev = sd(signal))

Y_sd = ts(q4_sd$standard_dev)

## Plot the standard deviations
autoplot(Y_sd) + geom_point(size = 3, shape = 1) +
        labs(x = "Season", y = "Signal Standard Deviation",
                      title = "Signal-to-Noise Ratio (SNR) for GPS")
```



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.