LAB MANUAL:01
MAT 3208: TIME SERIES

DATE:

EXPLORING RAW TIME SERIES | R

The most common first step when conducting time series analysis is to display your time series dataset in a visually intuitive format. The most useful way to view raw time series data in R is to use the **print()** command, which displays the **Start**, **End**, and **Frequency** of your data along with the observations.

Another useful command for viewing time series data in R is the **length()** function, which tells you the total number of observations in your data.
Some datasets are very long, and previewing a subset of data is more suitable than displaying the entire series. The **head(___, n =___)** and **tail(___, n =___)** functions, in which n is the number of items to display, focus on the first and last few elements of a given dataset respectively.

In this exercise, you'll explore the famous River Nile annual streamflow data, **Nile**. This time series dataset includes some metadata information. When calling **print(Nile)**, note that **Start = 1871** indicates that 1871 is the year of the first annual observation, and **End = 1970** indicates 1970 is the year of the last annual observation.

INSTRUCTIONS:

- Use the **print()** function to display the River Nile data. The data object is called Nile
- Use the **length()** function to identify the number of elements in your Nile dataset.
- Use **head()** to display the first 10 elements of the Nile dataset. To do so, set the n argument equal to 10.
- Use **tail()** to display the last 12 elements of the Nile dataset, again setting an appropriate value to the **n** argument.

LAB MANUAL:01
MAT 3208: TIME SERIES

BASIC TIME SERIES PLOTS

While simple commands such as **print()**, **length()**, **head()**, and **tail()** provide crucial information about your time series data, another very useful way to explore any data is to generate a plot.

In this exercise, you will plot the River Nile annual streamflow data using the **plot()** function. For time series data objects such as **Nile**, a **Time** index for the horizontal axis is typically included. From the previous exercise, you know that this data spans from 1871 to 1970, and horizontal tick marks are labeled as such. The default label of **"Time"** is not very informative. Since these data are annual measurements, you should use the label **"Year"**. While you're at it, you should change the vertical axis label to **"River Volume (1e9 m^{3})"**.

Additionally, it helps to have an informative title, which can be set using the argument **main**. For your purposes, a useful title for this figure would be "Annual River Nile Volume at Aswan, 1871-1970".

Finally, the default plotting type for time series objects is **"l"** for line. Connecting consecutive observations can help make a time series plot more interpretable. Sometimes it is also useful to include both the observations points as well as the lines, and we instead use **"b"** for both.

INSTRUCTIONS:

- Use **plot()** to display the Nile dataset.
- Use a second call to **plot()** to display the data, but add the additional arguments: **xlab = "Year", ylab = "River Volume (1e9 m^{3})".**
- Use a third call to **plot()** with your Nile data, but this time also add a title and include observation points in the figure by specifying the following arguments: **main = "Annual River Nile Volume at Aswan, 1871-1970", type ="b".**

LAB MANUAL:01
MAT 3208: TIME SERIES

WHAT DOES THE TIME INDEX TELL US?

Some data are naturally evenly spaced by time. The time series **discrete_data** shown in the top figure has 20 observations, with one observation appearing at each of the discrete time indices 1 through 20. Discrete time indexing is appropriate for **discrete_data.**

The time series **continuous_series** shown in the bottom figure also has 20 observations, it is following the same periodic pattern as **discrete_data**, but its observations are not evenly spaced. Its first, second, and last observations were observed at times 1.210322, 1.746137, and 20.180524, respectively. Continuous time indexing is natural for **continuous_series**, however, the observations are approximately evenly spaced, with about 1 observation observed per time unit. Let's investigate using a discrete time indexing for continuous_series.

INSTRUCTIONS:

- Use **plot(___, ___, type = "b")** to display **continuous_series** versus **continuous_time_index**, its continuous time index
- Create a vector 1:20 to be used as a discrete time index.
- Now use **plot(___, ___, type = "b")** to display **continuous_series** versus **discrete_time_index**
- Note the various differences between the resulting figures, but the approximation appears reasonable because the overall trend remained preserved