THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Time series Analysis and Modeling

DATS 6313

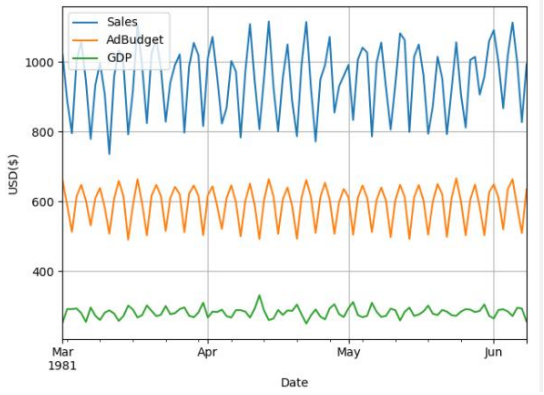
LAB # 1

Stationarity & Non-stationarity

Using the following Python libraries develop a program to answer the questions below:

```
import matplotlib.pyplot as plt
import pandas as pd
from statsmodels.tsa.stattools import adfuller
```

1- Load the time series data called 'tute1.csv' [the dataset can be found on the course GitHub]. This date relates to the quarterly sales for a small company over period 1981-2005. Sales contains the quarterly sales, AdBudget is the advertisement budget and GPD is the gross domestic product for a small company. Plot Sales, AdBudget and GPD versus time step in one graph. Add grid and appropriate title, legend to each plot. The x-axis is the time, and it should show the time (year). The y-axis is the USD($). The graph should be look like below.

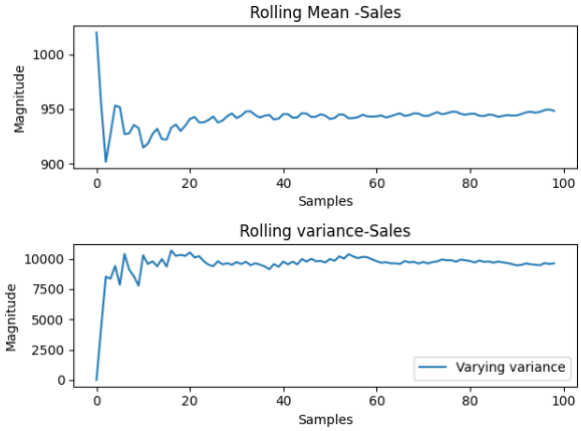


2- Find the time series statistics (average, variance, standard deviation, median) of Sales, AdBudget and GPD and display the Average, variance, and standard deviation as follow on the console:
   a. The Sales mean is : -- and the variance is : -- with standard deviation : ----median:----
   b. The AdBudget mean is : -- and the variance is : -- with standard deviation : -- median:----
   c. The GDP mean is :--- and the variance is : -------- with standard deviation : --- median:----

3- Prove that the Sales, AdBudget and GDP in this time series dataset is stationary. Hint: One way to show a process is stationary, is to plot the rolling mean and rolling variance versus number of samples which is accumulated through time. If the rolling mean and rolling variance stabilizes once all samples are included, then this is an indication that a data set is stationary. You need to plot the rolling mean and rolling variance in one graph using subplot [2x1] by creating a loop over the number of samples in the dataset and calculate the means & variances versus time. Plot all means and variances and show that the means and variances are almost constant. To perform this task, you need to create a loop with goes over number of observations in the dataset. During the first iteration, the first sample will load and the mean and variance will be calculated. During

the second iteration, the first two observations will load, and the mean and variance will be calculated and will append to the previous mean and variance. Repeat this process till the last observation is added the mean and variance will be calculated. You can use the following command to bring new data sample at each iteration. The plot the mean and variance over the time at the end. Save the above code under a function called 'Cal-rolling-mean-var '. You will use this function several times throughout the course. The Sales rolling mean and variance is shown below.

```
pd.read_csv('tute1.csv').head(i)# where is the number of samples
```



4- Write down your observation about the plot of the mean and variance in the previous step. Is Sales, GDP and AdBudget stationary or not? Explain why.

5- Perform an ADF-test to check if the Sales, AdBudget and GDP stationary or not (confidence interval 95% or above). Does your answer for this question reinforce your observations in the previous step? Hint: You can use the following code to calculate the ADF-test.

```
from statsmodels.tsa.stattools import adfuller

    def ADF_Cal(x):
        result = adfuller(x)

        print("ADF Statistic: %f" %result[0])
        print('p-value: %f' % result[1])
        print('Critical Values:')
        for key, value in result[4].items():
           print('\t%s: %.3f' % (key, value))
```

6- Perform an KPSS-test to check if the Sales, AdBudget and GDP stationary or not (confidence interval 95% or above). Does your answer for this question reinforce your observations in the previous steps? Hint: You can use the following code to calculate the KPSS-test.

```
from statsmodels.tsa.stattools import kpss

def kpss_test(timeseries):
    print ('Results of KPSS Test:')
    kpsstest = kpss(timeseries, regression='c', nlags="auto")
    kpss_output = pd.Series(kpsstest[0:3], index=['Test Statistic','p-value','Lags
Used'])
    for key,value in kpsstest[3].items():
        kpss_output['Critical Value (%s)'%key] = value
    print (kpss_output)
```

7- Repeat step 1-6 with 'AirPassengers.csv' on the GitHub. This timeseries dataset is univariate with #passengers as an attribute.

8- If the #passengers is not stationary, it needs to become stationary by transformation.
   a. Perform a 1st order non-seasonal differencing. Is the dataset become stationary? Explain why.
   b. Perform a 2nd order non-seasonal differencing. Is the dataset become stationary? Explain why.
   c. Perform a 3rd order non-seasonal differencing. Is the dataset become stationary? Explain why.
   d. If the procedures in steps a, b and c does not make the dataset stationary then perform a log transformation of the original raw dataset followed by a 1st order differencing then plot the rolling mean and variance. Perform ADF-test and KPSS-test on the transformed dataset and display the results on the console. This step should make the dataset stationary which means the rolling mean and variance is stabilize and the ADF-test confirms stationarity.

Submission guidelines:

A. The softcopy of the developed Python code .py must also be submitted separately. Please make sure the developed python code runs without any error by testing it through PyCharm software. **The developed python code with any error will subject to 50% points penalty.**
B. Add an appropriate x-label, y-label, legend, and title to each graph.
C. Write a report and answer all the above questions. Include the required graphs into your report.