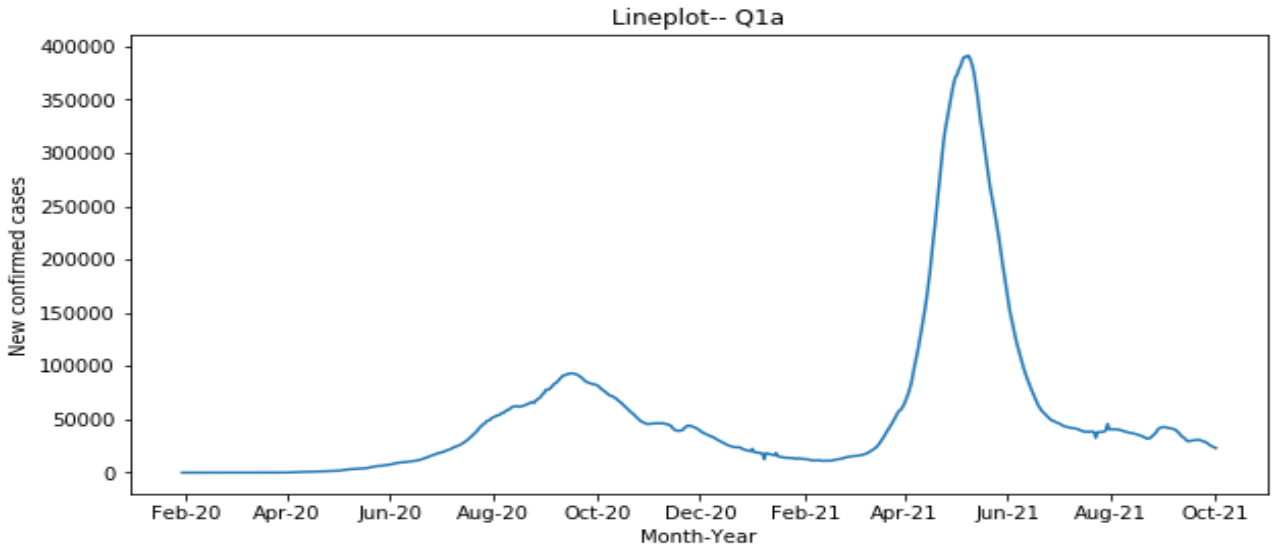
Lab6: Autoregression

Deadline for submission: 3 November 2021

You are given a dataset containing details about **new COVID-19 cases recorded in India on daily basis** as a csv file (`daily_covid_cases.csv`). It shows the rolling 7-day average of newly confirmed cases starting from 30th-Jan-2020 to 2-Oct-2021. Rows are indexed with dates; the first column represents the date and the second column represents the new COVID-19 cases recorded that day. We will use this dataset to build an autoregression (AR) model.

1. Autocorrelation line plot with lagged values:
   a. Create a line plot with the x-axis as index of the day and y-axis as the number of Covid-19 cases, as given below. Observe the first wave (around August-2020) and second wave (May-2021) of COVID-19 in India.



   b. Generate another time sequence with a one-day lag to the given time sequence. Find the Pearson correlation (autocorrelation) coefficient between the generated one-day lag time sequence and the given time sequence.

   c. Generate a scatter plot between the given time sequence and one-day lagged generated sequence in 1.b. What do you infer regarding correlation? Does it match with the computed correlation coefficient in 1.b?

   d. Generate multiple time sequences with different lag values (1-day, 2-days, 3-days up to 6-days). Compute the Pearson correlation coefficient between each of the generated time sequences and the given time sequence. Create a line plot between obtained correlation coefficients (on the y-axis) and lagged values (on the x-axis).

   e. Plot a correlogram or **A**uto **C**orrelation **F**unction using python inbuilt function '`plot_acf`'. Observe the trend in the line plot with increase in lagged values and relate with that of 1 d.

2. A general autoregression (AR) model estimates the unknown data values as a linear combination of given lagged data values. For example, data value at (t+1) instant, denoted by $x(t+1)$, can be estimated from its previous instance values, such as $x(t+1) = w_0 + w_1*x(t) + w_2*x(t-1) + \ldots + w_p*x(t-p+1)$. The coefficients w_0, w_1, … w_p can be estimated while training the autoregression model on training dataset.

a. Split the data into two parts. The initial 65% of the sequence for training data and the remaining 35% of the sequence as test data. (*You may use slicing operation for the same to maintain the order of the sequence. Note that, you should not shuffle randomly*.) This test set approximately covers the second wave of COVID-19. Plot the train and test sets. Generate an autoregression (AR) model using `AutoReg()` function from `statsmodels` library. This function generates an AR model with the specified training data and lagged values (given as its input). Use 5 lagged values as its input (p=5). Train/Fit the model onto the training dataset. Obtain the coefficients (w_0, w_1, ..., w_p) from the trained AR model.

Train
Tr1 Tr2 ... TrT
Tst1 Tst2 ... TstN
Test
I/p | predict
TrT | Test1
Tst1 | Test2
Tst2 | Test3
:
Tst N-1 | Test N

b. Using these coefficients, predict the values (using the relation given above) for the test dataset. Note that, you have to make a 1-step ahead prediction each time. An example code snippet is given below (**Code snippet to train AR model and predict using the coefficients**).

   i. Give a scatter plot between actual and predicted values.

   ii. Give a line plot showing actual and predicted test values.

   iii. Compute RMSE (%) and MAPE between actual and predicted test data.

3. Generate five AR models using `AutoReg()` function with lag values as 1, 5, 10, 15 and 25 days. Compute the RMSE (%) and MAPE between predicted and original test data values in each case. Give a bar chart showing RMSE (%) on the y-axis and lagged values on the x-axis. Also, give a bar chart showing MAPE on the y-axis and lagged values on the x-axis. Infer the changes in RMSE (%) and MAPE with changes in lagged values.

4. Compute the heuristic value for the optimal number of lags up to the condition on autocorrelation such that *abs*(*AutoCorrelation*) > *2/sqrt*(*T*), where *T* is the number of observations in training data. Use it as input in `AutoReg()` function to predict the new COVID-19 cases on daily basis and compute the RMSE(%) and MAPE value. Compare this result with that of Question 3.

Extra Work.:
Take the full data (Jan 30 2020 to Oct 2, 2021)
Build AR model for best p-lag [Observed in Qn. 4]
- Predict newcases for Oct 3 taking Oct 2 value as I/p
- Predict newcases for Oct 4 taking "predicted value" of Oct 3 as i/p
- Do it till Jan 31, 2022
- Can you observe any peak?

Functions/Code Snippets:

```
from statsmodels.tsa.ar_model import AutoReg as AR

# Train test split
series = pd.read_csv('daily_covid_cases.csv',
                     parse_dates=['Date'],
                     index_col=['Date'],
                     sep=',')
test_size = 0.35 # 35% for testing
X = series.values
tst_sz = math.ceil(len(X)*test_size)
train, test = X[:len(X)-tst_sz], X[len(X)-tst_sz:]
```

 # Code snippet to train AR model and predict using the coefficients.

```
Window = 1 # The lag=1
model = AR(train, lags=window)
model_fit = model.fit()  # fit/train the model
coef = model_fit.params  # Get the coefficients of AR model

#using these coefficients walk forward over time steps in test, one
step each time
history = train[len(train)-window:]
history = [history[i] for i in range(len(history))]
predictions = list() # List to hold the predictions, 1 step at a time
for t in range(len(test)):
    length = len(history)
    lag = [history[i] for i in range(length-window,length)]
    yhat = coef[0]  # Initialize to w0
    for d in range(window):
        yhat += coef[d+1] * lag[window-d-1]  # Add other values
    obs = test[t]
        predictions.append(yhat)  #Append predictions to compute RMSE
 later
    history.append(obs)  # Append actual test value to history, to be
used in next step.
```

Instructions:

- **Your python program(s) should be well commented. The comment section at the beginning of the program(s) should include your name, registration number and mobile number.**
- **The python program(s) should be in the file extension .py**
- **The Report should be strictly in PDF form. Write the report in word or latex form and then convert to PDF form. Template for the report (in word and latex) is uploaded.**
- **The first page of your report must include your name, registration number and mobile number. Use the template of the report given in the assignment.**
- **Upload your program(s) and report in a single zip file. Give the name as <roll_*number*>_Assignment6.zip. Example: *b20001*_Assignment6.zip**
- **Upload the zip file in the link corresponding to your group only.**

In case the program found to be copied from others, both the person who copied and who helped will get zero as a penalty.