Time Series Modeling

Seasonal ARIMA Model

Contents

Case Study

Background

- Annual Sales for a specific company from year 1961 to 2017

Objective

- To assess stationarity of time series

Available Information

- Number of cases: 57
- Variables: Year, sales(in 10's GBP)

Data Snapshot

turnover_annual data

Variables

Observations on Discrete Time Scale

Year	sales
1961	224786
1962	230034
1963	236562
1964	250960
1965	261615
1966	268316
1967	283589
1968	280160
1969	301422
1970	308018
[illegible]	[illegible]
1974	364834
1975	392503

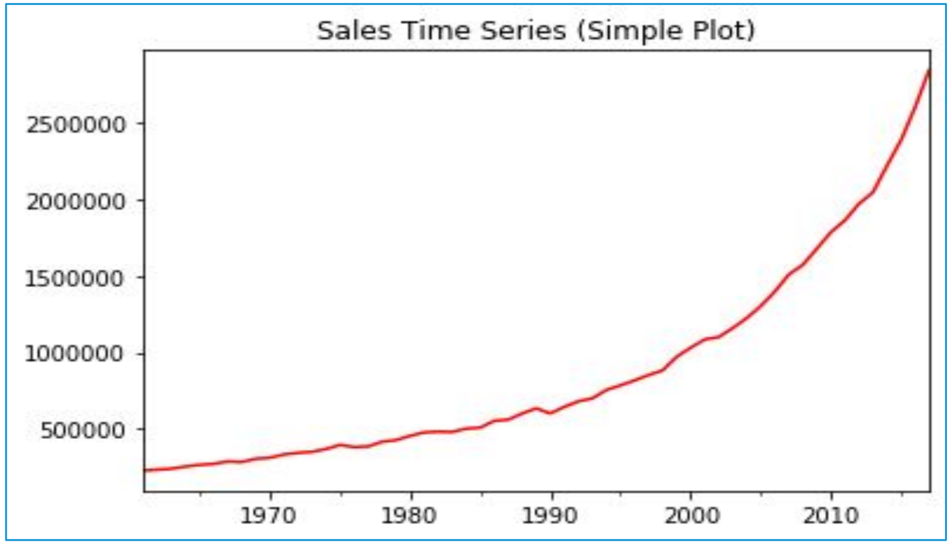
Columns	Description	Type	Measurement	Possible values
Year	Financial Year	Numeric	-	-
sales	sales(in 10's GBP)	Numeric	In British Pound	Positive values

Creating and Plotting Time Series in Python

```
#Importing turnover_annual data
import pandas as pd
salesdata=pd.read_csv('turnover_annual.csv')
#Creating and Plotting a Time Series Object
rng = pd.date_range('01-01-1961','31-12-2017',freq='Y')
s = salesdata.sales.values
salesseries = pd.Series(s, rng)

salesseries.plot(color='red', title ="Sales Time Series
(Simple Plot)")
```

- **date_range()** creates pandas date object.
- **freq ='Y' indicates yealy data**
- **pd.Series()** creates time series object
- Plot function gives line chart



Interpretation :

- The time-series clearly shows a positive trend.

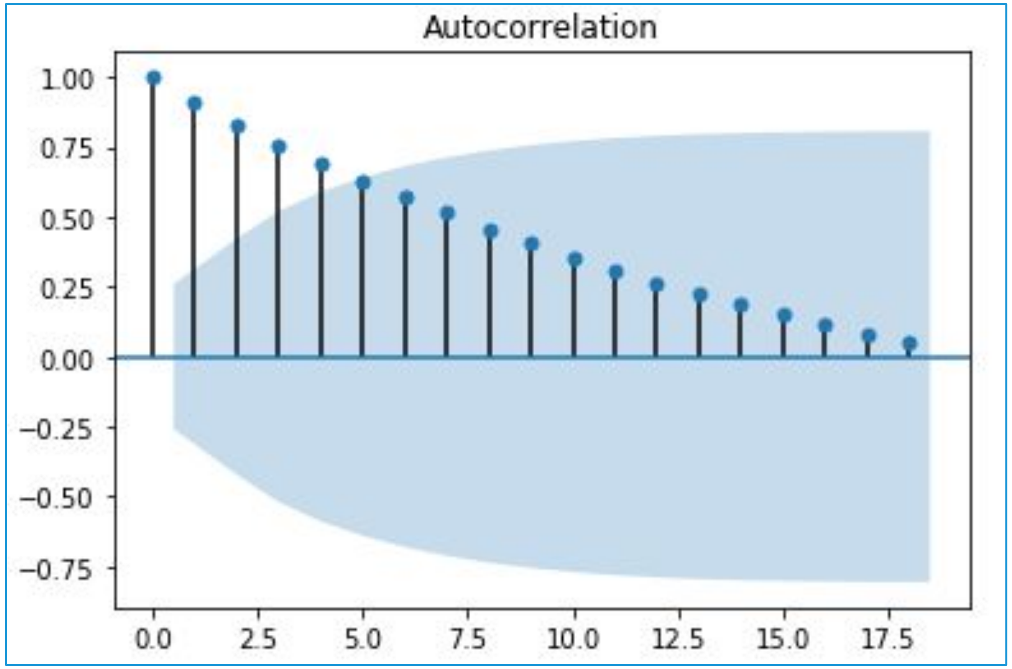
Checking Stationarity – Correlogram

ACF Plot

```
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(salesseries)
```

- **plot_acf()** returns an ACF (Auto Correlation Function) plot.

Output



Interpretation :

- We can observe that there is a very slow decay which is a sign of Non-stationarity.

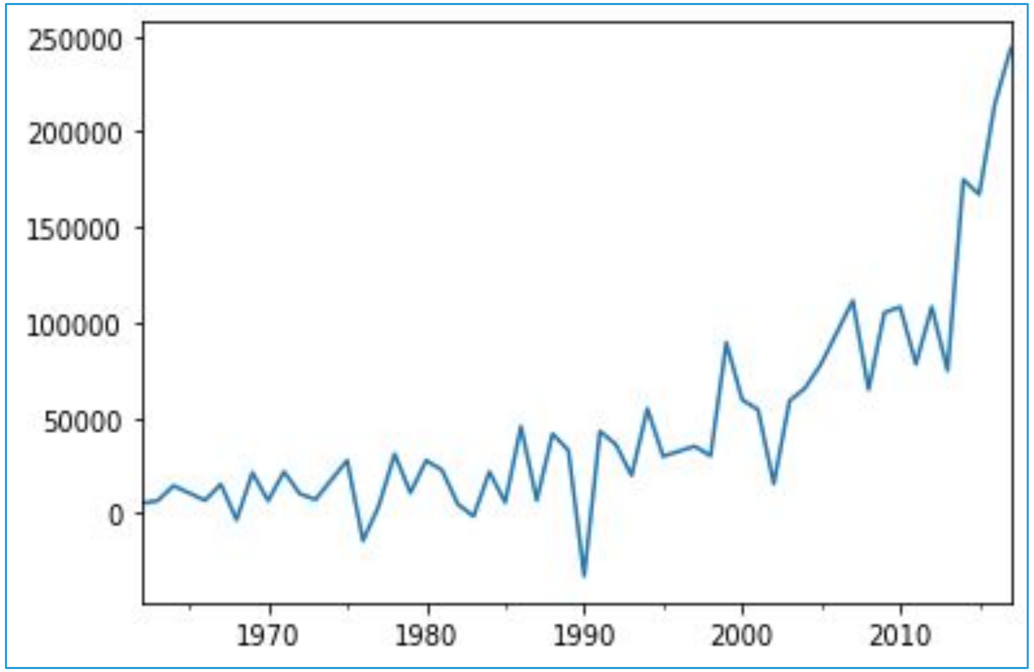
Plot of 1st Order Differenced Time Series

```
# Creating and Plotting a Difference Series
```

```
from statsmodels.tsa.statespace.tools import diff
salesdiff = diff(salesseries)
salesdiff.plot()
```

- diff() gives 1st order differences
- plot function gives line chart for differenced series

```
# Output
```



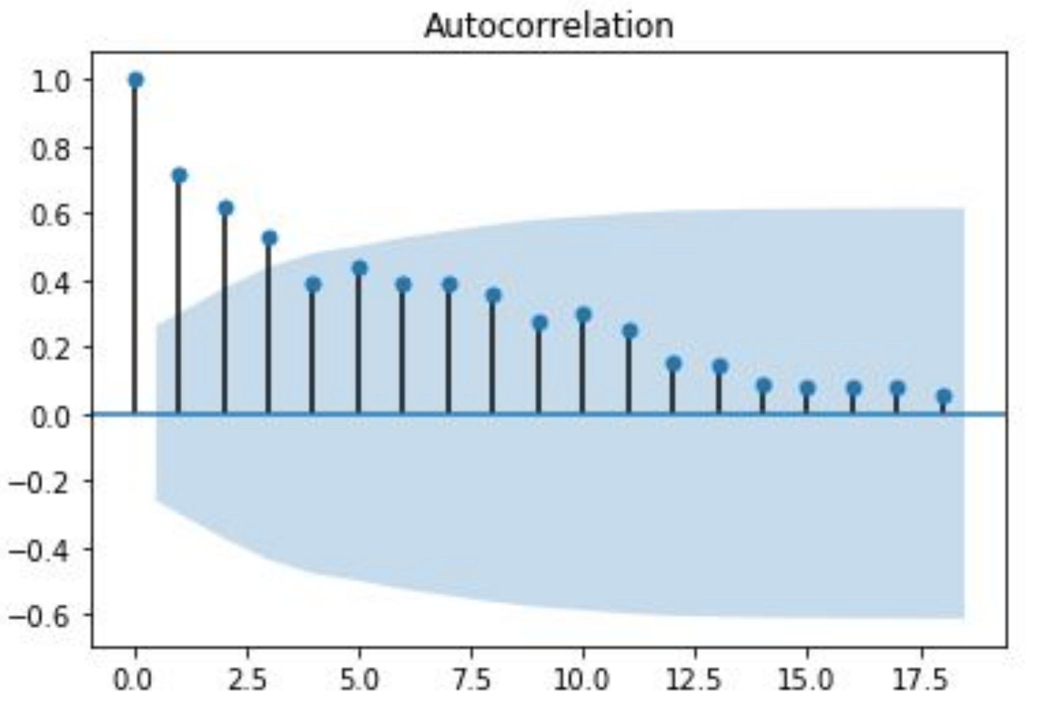
Interpretation :

- Even after first order differencing, the series looks non-stationary.

Correlogram for 1st Order Differenced Time Series

```
# ACF Plot
plot_acf(salesdiff)
```

```
# Output
```



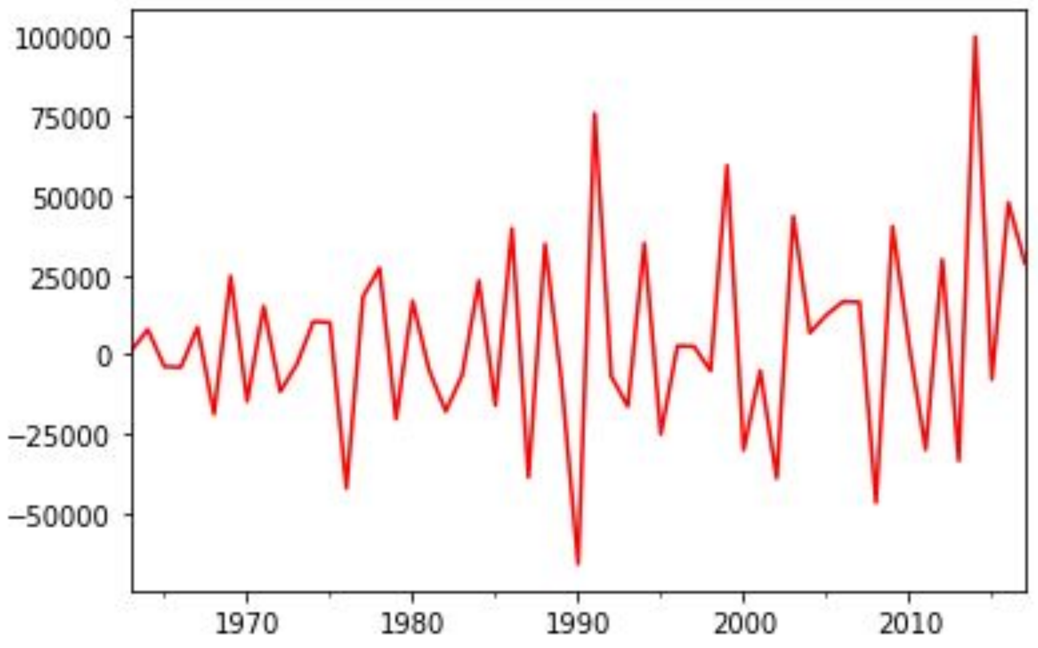
Interpretation :

- ACF plot shows slow decay
- Stationarity is not achieved with first difference.

Plot of 2nd Order Differenced Time Series

```
#Creating and Plotting 2nd Difference Series
salesdiff2 = diff(salesdiff)
salesdiff2.plot(color='red')
```

```
# Output
```

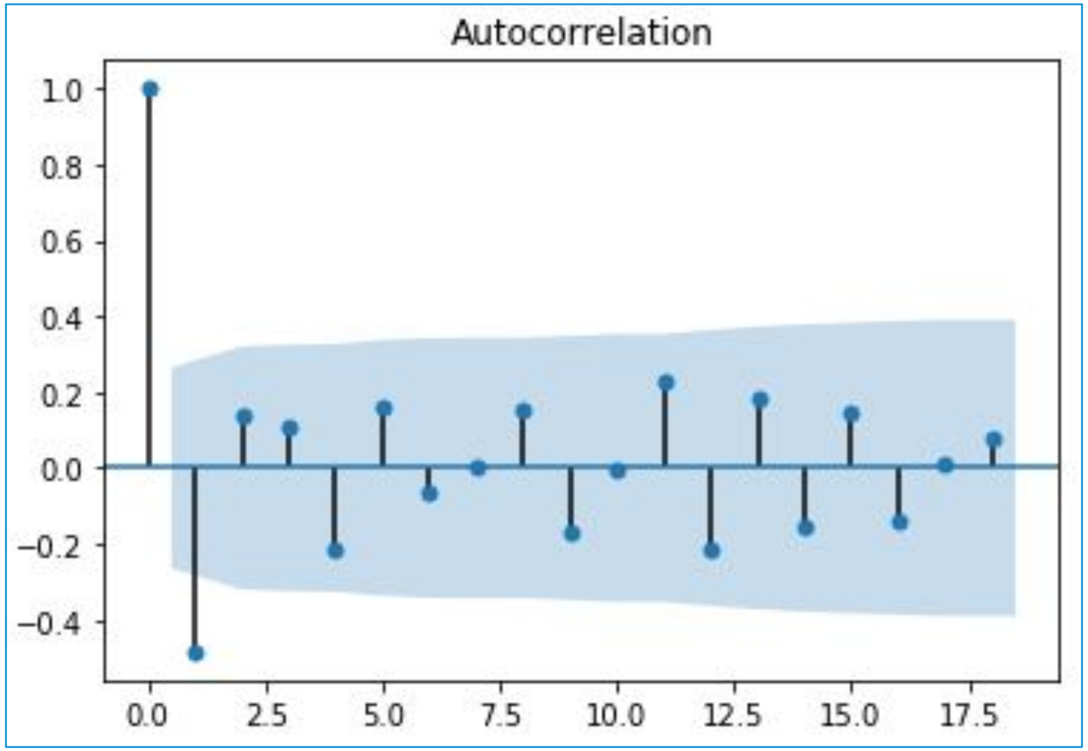


Interpretation :

- After 2nd order differencing, the series looks **stationary**.

Correlogram for 2nd Order Differenced Time Series

```
# ACF Plot
plot_acf(salesdiff2)
# Output
```



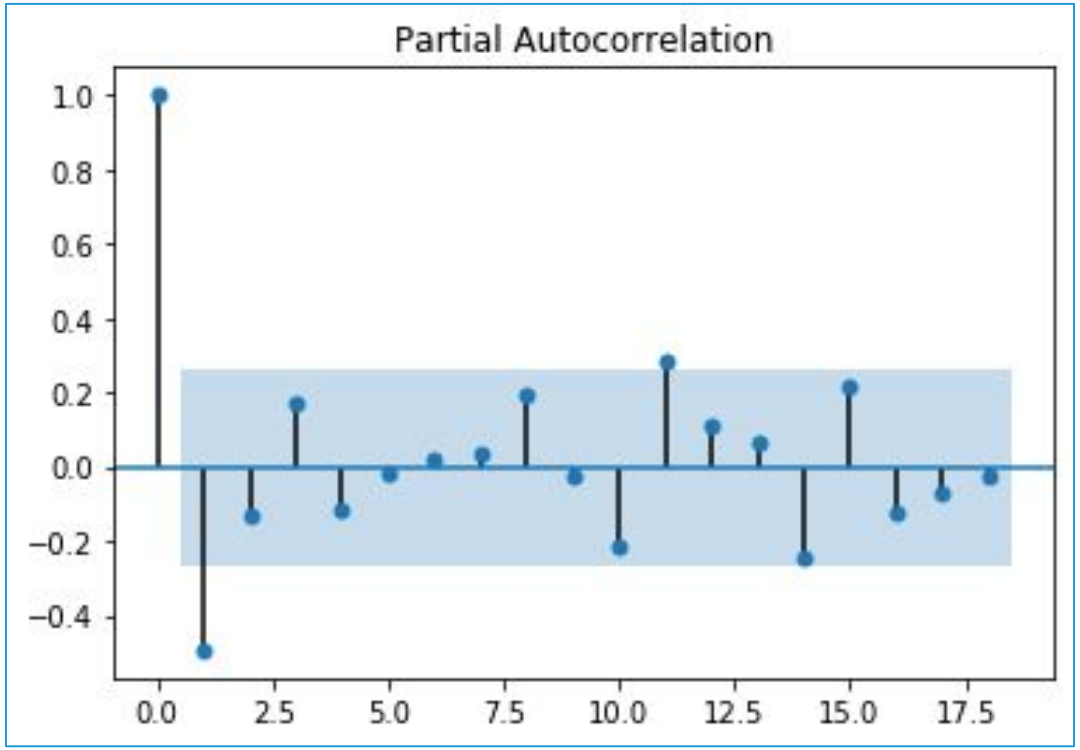
Interpretation :

- Stationarity is achieved with 2nd order difference.

Partial Autocorrelations for 2nd Order Differenced Time Series

```
# PACF Plot
plot_pacf(salesdiff2)
# Output
```

- **plot_pacf()** returns an PACF (Partial Auto Correlation Function) plot.



Interpretation :

- Stationarity is achieved with 2nd order difference.

Analytical Method – Dickey Fuller (DF) Test

- A linear stochastic process has a unit root if 1 is the root of the process's characteristic equation. Such a process is non-stationary.
- Dickey and Fuller pioneered idea of testing for unit roots for stationarity checking.

Consider X_t (t=1,2,3,...) is a time series of the form

$$X_t=\rho X_{t-1}+U_t \quad \ldots\ldots\ldots\ldots\ldots\ldots(1)$$

If $\rho=1$ then X_t becomes a random walk

- We assume that $U_t \sim IID(0, \sigma^2)$, i.e U_t is a white noise
- Therefore, we are interested in testing for $\rho=1$

* IID means independent and identically distributed

Dickey Fuller (DF) Unit Root Test

Objective	To test the **null hypothesis** that **time series is not stationary**

Null Hypothesis H_0: $\rho=1$

Alternate Hypothesis H_1: $\rho< 1$

$(X_t-X_{t-1}) =\Delta X_t=(\rho-1)\ X_{t-1}+U_t$..............from (1)

H_0: $\rho^*=0$, H_1: $\rho^*<0$, $\rho^*=(\rho-1)$

Test Statistic	$(\rho^*/SE(\rho^*))$ Test statistic follows DF distribution under null
Decision Criteria	Reject the null hypothesis **tcal < DF table value**

Dickey Fuller Test

```
# Install "arch"
pip install arch
# Import "ADF" from library "arch"
from arch.unitroot import ADF

adf = ADF(salesseries,lags=0,trend='nc')
adf.summary()
```

- **ADF()** performs a Dickey Fuller unit root test on time series data.
- **lags=** allows to mention the number of lags to use in the ADF regression. We have used zero.
- **trend='nc'** specifies no trend and constant in regression

```
# Output
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                 19.275
P-value                         1.000
Lags                                0
-------------------------------------

Trend: No Trend
Critical Values: -2.61 (1%), -1.95 (5%), -1.61 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Interpretation :

- Time series is non-stationary as value of test statistic is greater than 5% critical value.

Dickey Fuller Test

```
# Checking stationarity for series with difference of order 2
```

```
adf = ADF(salesdiff2,lags=0,trend='nc')
adf.summary()
```

```
# Output
```

```
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                -11.908
P-value                         0.000
Lags                                0
-------------------------------------

Trend: No Trend
Critical Values: -2.61 (1%), -1.95 (5%), -1.61 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Interpretation :

- Time series is stationary as value of test statistic is less than 5% critical value.

Quick Recap

Correlograms	• **`plot_acf() & plot_pacf()`** function in Python generate Correlograms
Differencing a Time Series	• Simple numeric function **`diff()`** can be used to difference a series
Dickey Fuller Test	• **`ADF()`** function from the package **`arch`** performs a Dickey Fuller test • The output gives test statistic and critical values for the test statistic