Aim: Time Series Analyisis in Python

Thoery: Time series data is a sequential arrangement of data points organized in consecutive time order. Time-series analysis consists of methods for analyzing time-series data to extract meaningful insights and other valuable characteristics of the data.

Time-series data analysis is becoming very important in so many industries, like financial industries, pharmaceuticals, social media companies, web service providers, research, and many more. To understand the time-series data, visualization of the data is essential. In fact, any type of data analysis is not complete without visualizations, because one good visualization can provide meaningful and interesting insights into the data.

Time series data can be classified into two types:

1. Continuous Time Series Data: Involves measurements recorded at regular intervals, with a continuous range of values. Examples include:
    - Temperature data (e.g., hourly or daily)
    - Stock market data (e.g., prices throughout trading hours)
    - Sensor data (e.g., pressure or air quality readings)
2. Discrete Time Series Data: Consists of distinct, separate data points, often limited to specific values or categories. Examples include:
    - Count data (e.g., number of occurrences)
    - Categorical data (e.g., customer segments)
    - Binary data (e.g., two possible outcomes)

Time Series Decomposition:

Time series decomposition is a method of breaking down a time series into its components (trend, seasonality, and residual noise). A simple decomposition can be expressed as:

$Yt=Tt+St+Et$

Where:

- Yt is the observed value at time t,
- Tt is the trend component,
- St is the seasonal component,
- Et is the residual error or noise.

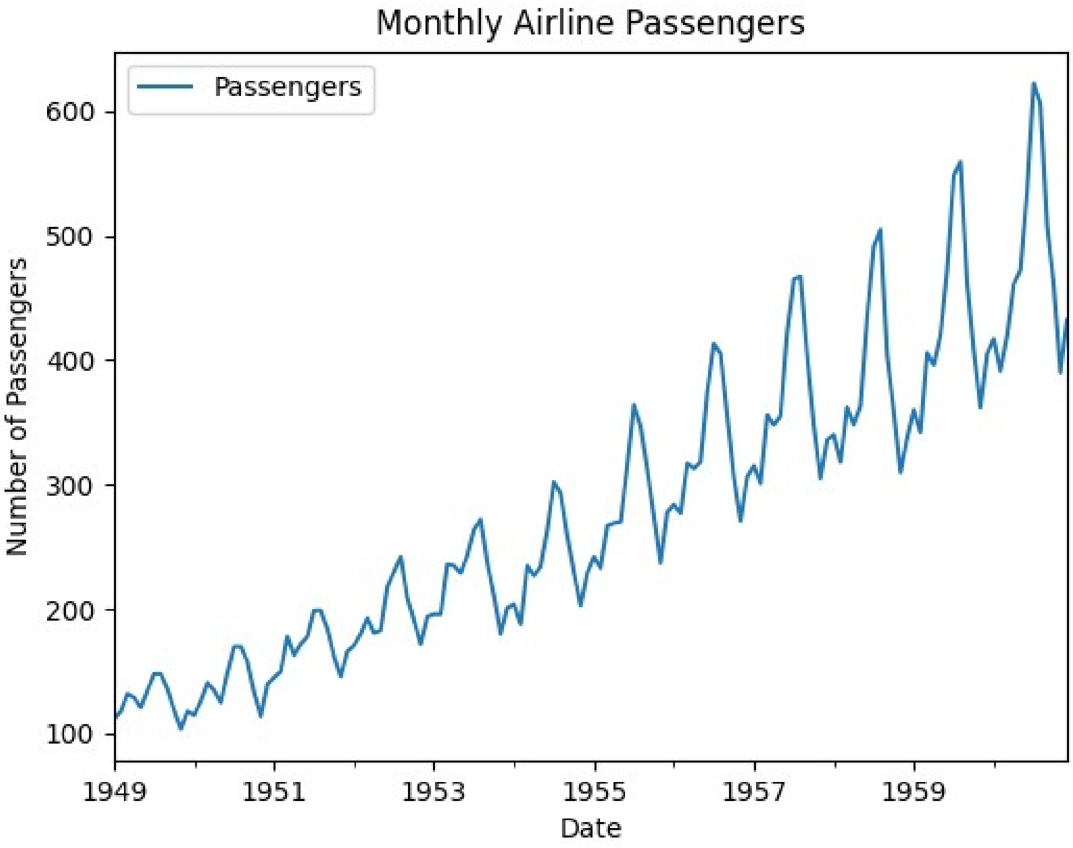
Code:

```
import pandas as pd

import numpy as np
```

```
import matplotlib.pyplot as plt
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/airline-passengers.csv"
data = pd.read_csv(url, header=0, parse_dates=[0], index_col=0, date_parser=pd.to_datetime)

data.plot()
plt.title("Monthly Airline Passengers")
plt.xlabel("Date")
plt.ylabel("Number of Passengers")
plt.show()

result = seasonal_decompose(data, model='multiplicative', period=12)
result.plot()
plt.show()

def adf_test(series):
    result = adfuller(series, autolag='AIC')
    print(f"ADF Statistic: {result[0]}")
    print(f"p-value: {result[1]}")
    if result[1] <= 0.05:
        print("Series is stationary")
    else:
        print("Series is not stationary")
adf_test(data)
model = SARIMAX(data, order=(1,1,1), seasonal_order=(1,1,0,12))
model_fit = model.fit(disp=False)

forecast = model_fit.forecast(steps=12)
```

```
print(f"Forecasted values: \n{forecast}")

plt.plot(data, label='Historical Data')
plt.plot(pd.date_range(start=data.index[-1], periods=13, freq='M')[1:], forecast, label='Forecast', color='red')
plt.title("Airline Passengers Forecast")
plt.xlabel("Date")
plt.ylabel("Number of Passengers")
plt.legend()
plt.show()
```

Output:



Conclusion:

Time-series analysis is critical for forecasting future values and understanding underlying trends, seasonality, and patterns in data. Whether it's for stock market prediction, sales forecasting, or understanding environmental trends, time series analysis provides valuable insights for decision-making.