Lecture 9

Time Series

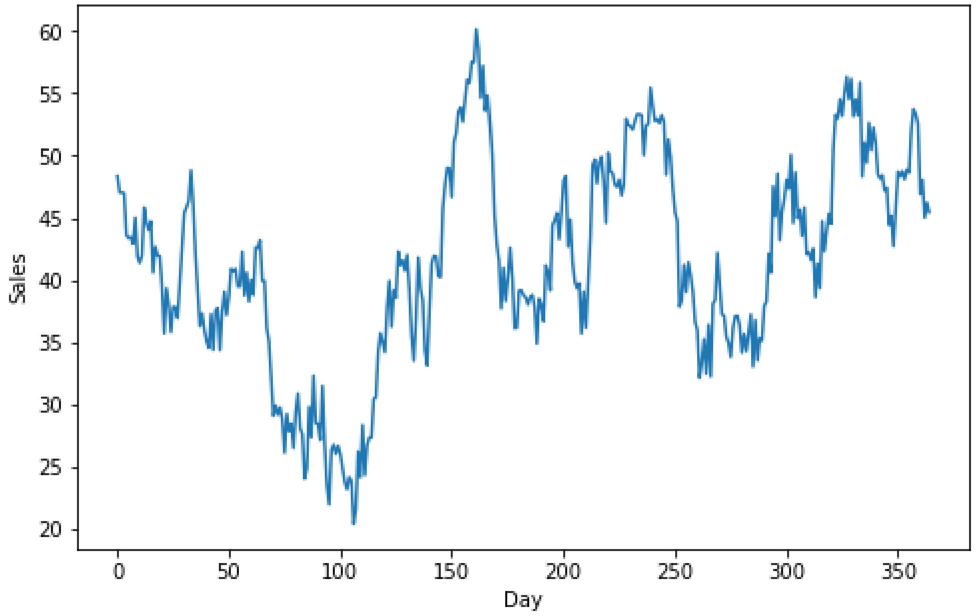
Basics of time series

- Random walks
- Autoregressive series
- Moving average series
- Autocorrelation and partial autocorrelation
- Differencing
- Seasonality

Example

My cookie factory wants to predict how many cookies will be sold in a day

We have recorded the number of sales every day for one year



Random Walks

- A random walk is a process in which each step is randomly determined

- The simplest random walk is an autoregressive random walk

$$X_t = X_{t-1} + \epsilon_t$$

Autoregressive Models

- An autoregressive (AR) is a generalization of random walks in which the value at time t depends on multiple preceding values

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \epsilon$$

- AR models typically do not skip lags. A model with three lags – AR(p) – can then be written as

$$X_t = \sum_{i=1}^{p} \phi_i X_{t-i}$$

Moving Average Models

- A moving average model has a value that depends on the last few error terms ϵ

- Moving average models – MA(q) – have equations of the form

$$X_t = \sum_{i=1}^{q} \epsilon_{t-i}$$

ARMA Models

- Models can have both autoregressive and moving average terms. These are called ARMA models and are parameterized by (p,q)

$$X_t = \sum_{i=1}^{p} \phi_i X_{t-1} + \sum_{j=1}^{q} \epsilon_{t-1} + \epsilon_t$$

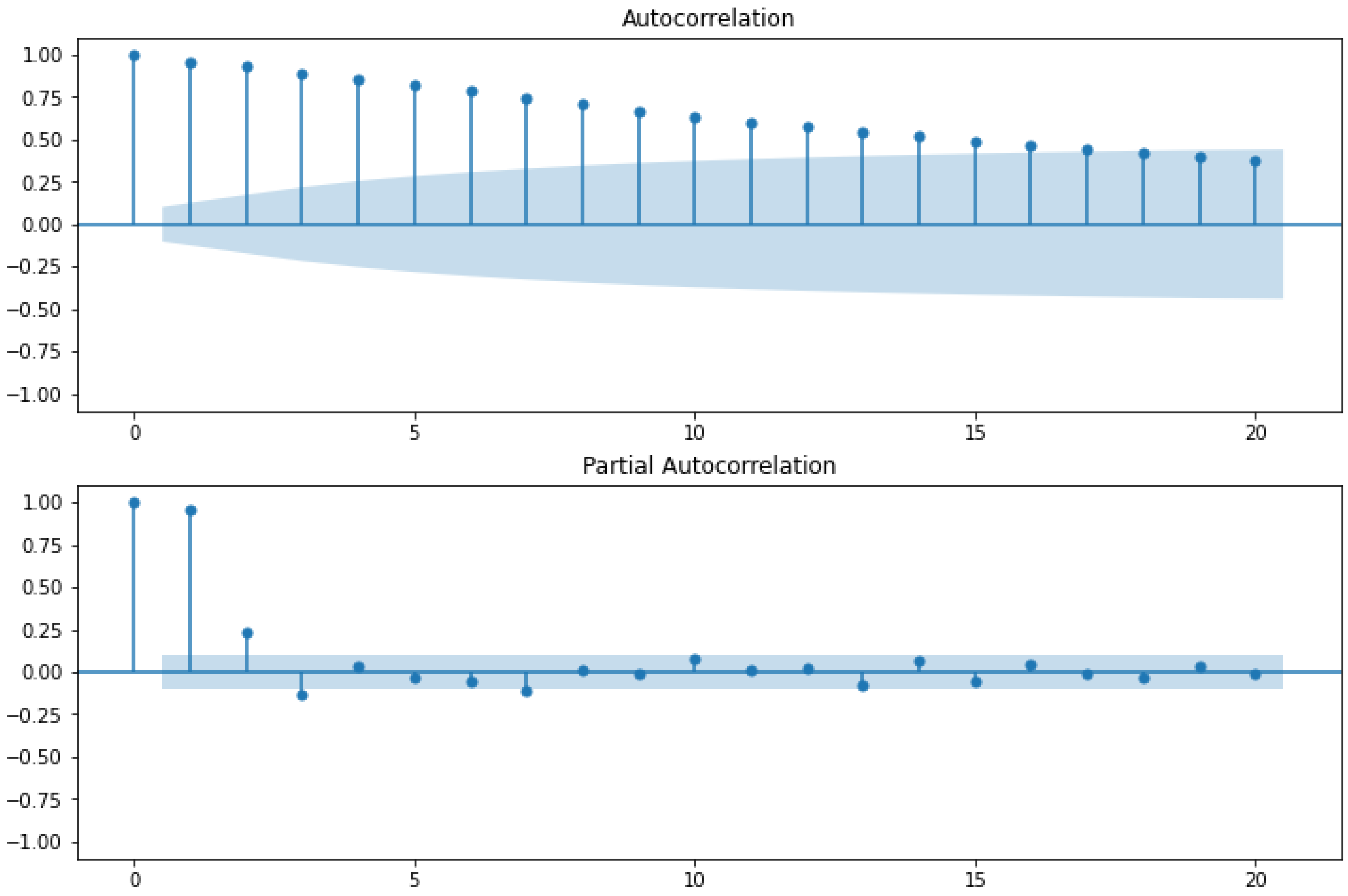
- To select the lags *p* and *q*, we can look at the autocorrelation and partial autocorrelation

Autocorrelation & Partial Autocorrelation

```
fig = plt.figure(figsize=(12, 8))
ax1 = fig.add_subplot(211)

fig = sm.graphics.tsa.plot_acf(sales, lags=20, ax=ax1)
ax2 = fig.add_subplot(212)

fig = sm.graphics.tsa.plot_pacf(sales, lags=20, ax=ax2)
```

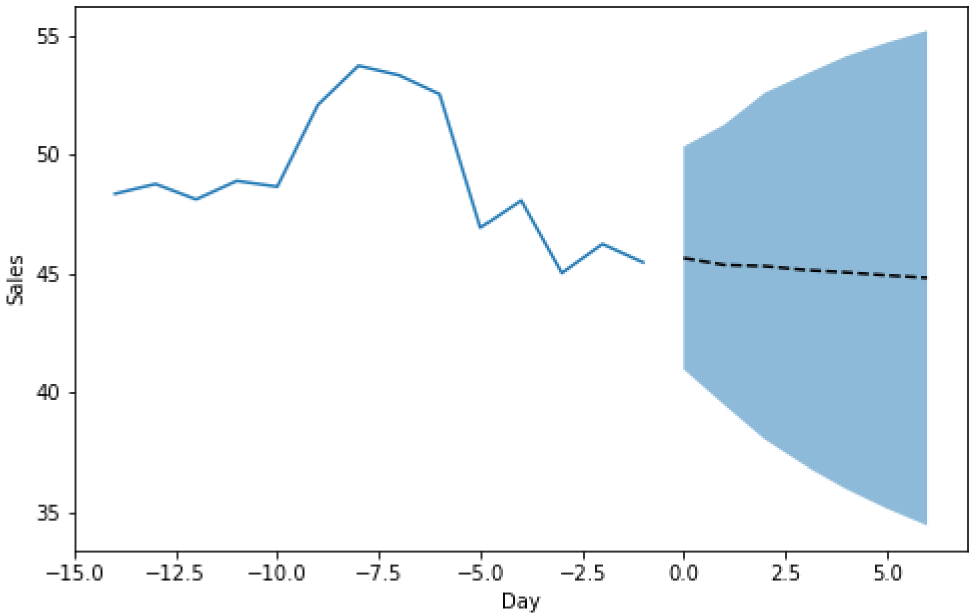


Fitting Using Statsmodels

```
arma = sm.tsa.arima.ARIMA(data['Power'], order=(4,0,0)).fit()
arma.summary()
```

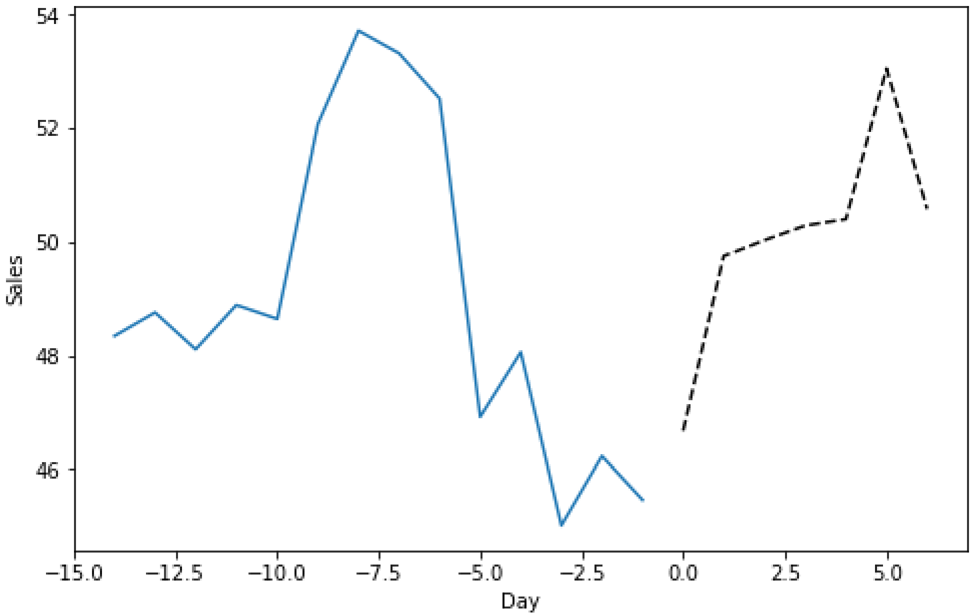
```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  365
Model:                 ARIMA(3, 0, 0)   Log Likelihood                -836.722
Date:                Tue, 12 Mar 2024   AIC                           1683.444
Time:                        13:40:58   BIC                           1702.943
Sample:                             0   HQIC                          1691.193
                                - 365
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         42.1426      2.929     14.387      0.000      36.401      47.884
ar.L1          0.7633      0.053     14.356      0.000       0.659       0.867
ar.L2          0.3338      0.065      5.161      0.000       0.207       0.461
ar.L3         -0.1388      0.052     -2.657      0.008      -0.241      -0.036
sigma2         5.6971      0.453     12.564      0.000       4.808       6.586
===================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                 0.65
Prob(Q):                              0.91   Prob(JB):                         0.72
Heteroskedasticity (H):               1.04   Skew:                             0.04
Prob(H) (two-sided):                  0.85   Kurtosis:                         2.81
===================================================================================
```

Using the results



Short-Term Forecasting

Predict values over the next few time periods



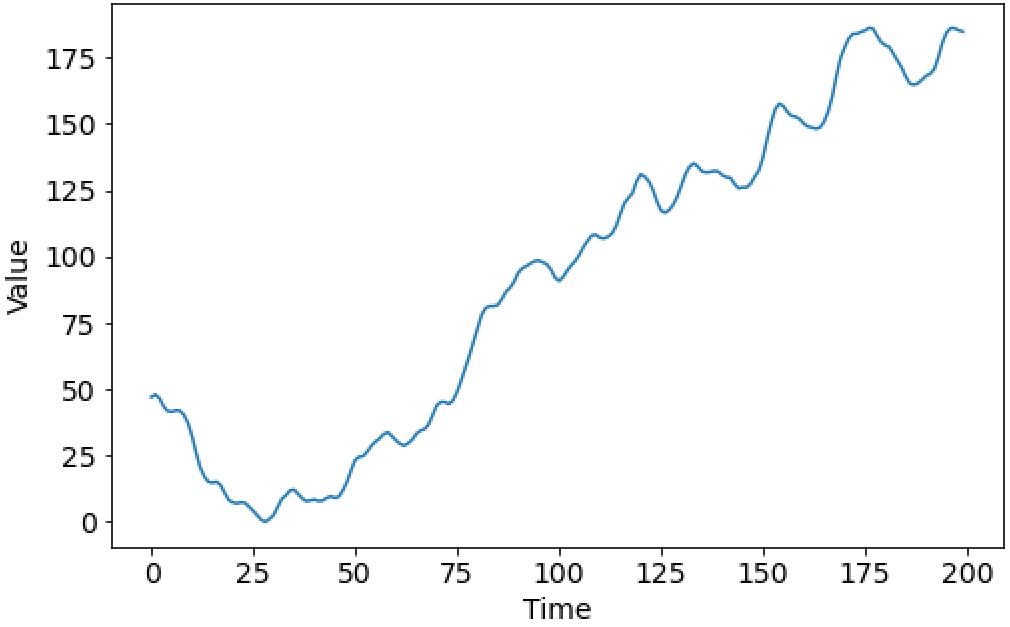
Representative Scenarios

Create sample time series following the same patterns

Stationary and Non-Stationary Series

- ARMA models assume the time series is stationary, meaning that the distribution of values does not change as time passes

- What happens if we do not have a stationary time series?



Non-Stationary Time Series

- We can check if a time series is stationary using the Augmented Dickey-Fuller (ADF or ADFuller) test
  - We won't go into details on how this is calculated, but it ends with a hypothesis test that gives a p-value. Lower p-value means more stationary

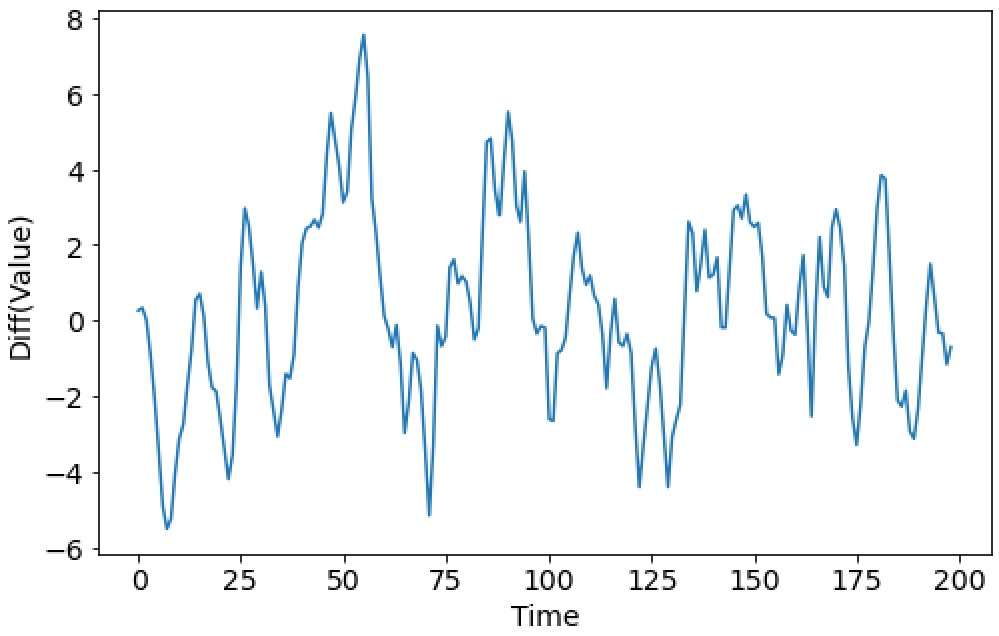
```
sm.tsa.adfuller(ts)
>>> (0.18680600896643582, # Test statistic
       0.9715117981353568, # P-Value
       0, 99, # Lags used, observations
       {'1%': -3.498198082189098, # critical values
       '5%': -2.891208211860468,
       '10%': -2.5825959973472097},
    526.2316331583049) # best IC score
```

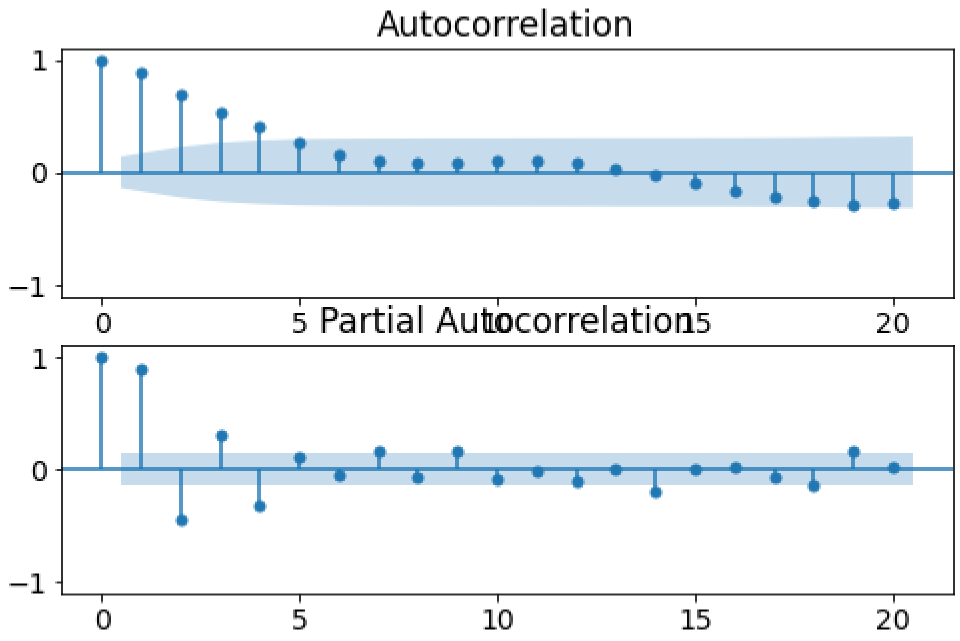
Non-Stationary Time Series

- For non-stationary time series, we can introduce one or more differences to the ARMA model – creating an ARIMA model
  - An ARIMA(p,1,q) model is equivalent to an ARMA(p,q) model on the differences of the time series

- You can estimate terms for *p* and *q* using ACF/PACF plots of the **difference** in the time series

Non-Stationary Time Series

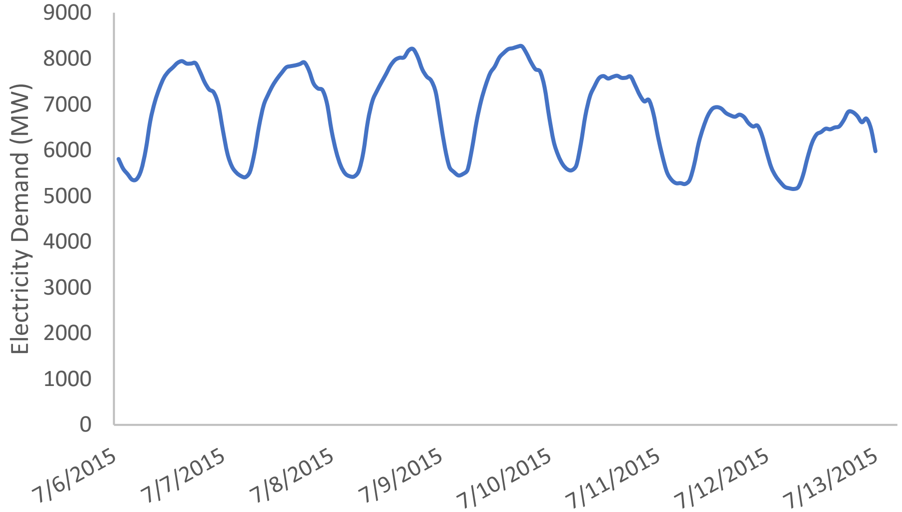




Seasonality

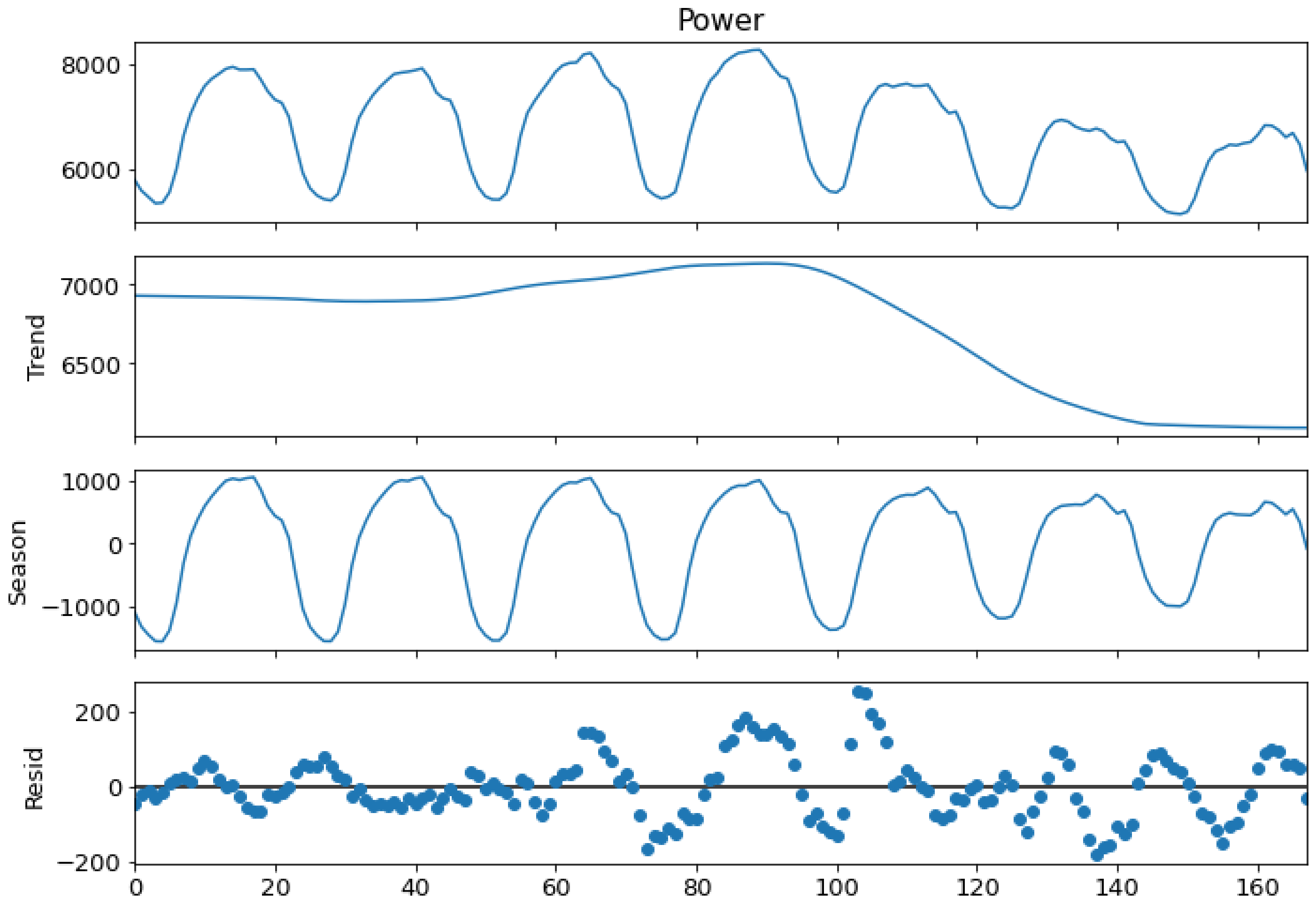
- Seasonality is when time series data follows a recurring pattern over some interval

- We can decompose seasonal data into seasonal, trend, and residual terms
  - Seasonal – repeating pattern
  - Trend – long-term average
  - Residual – leftovers



STL Decomposition

- **S**easonal-**T**rend Decomposition by **L**oess uses smoothing estimates to generate the component time series




```
from statsmodels.tsa.seasonal import STL
stl = STL(df['Power'], period=24)
result = stl.fit()
```

STL Decomposition

- The residuals of an STL decomposition can be treated as an ARIMA series

```
from statsmodels.tsa.seasonal import STL
stl = STL(df['Power'], period=24)
result = stl.fit()

res = sm.tsa.ARIMA(result.resid,
                   order=(1,0,1)).fit()
```