

Introduction to Computer Programming

Week 9.2: Curve Fitting



In [43]:

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 %matplotlib inline
```

Fitted function

Example 1:

Fit a first degree polynomial (linear function) to the x, y data.

Print the coefficients of the fitted function.

In [44]:

```
1 x = np.array([1, 6, 3, 4, 10, 2, 7, 8, 9, 5])
2 y = np.array([2, 4, 5, 4, 13, 3, 4, 8, 12, 4])
3
4 c1 = np.polyfit(x, y, 1) # coefficients of 1st degree fitted po
5
6 print(c1)
7 print(c1[0], c1[1])
```

```
[ 1.07272727e+00 -5.61733355e-15]
1.0727272727272728 -5.61733354972272e-15
```

Try it yourself

Example 2:

Fit a second degree polynomial to the x, y data.
(Remember to import numpy to use `polyfit`).

Print the coefficients of the fitted function.

```
In [46]: 1 x = np.array([1, 6, 3, 4, 10, 2, 7, 8, 9, 5])
          2 y = np.array([2, 4, 5, 4, 13, 3, 4, 8, 12, 4])
          3
          4 c2 = np.polyfit(x, y, 2) # 2nd degree poly
          5
          6 print(c2[0], c2[1], c2[2])
```

0.19318181818181812 -1.0522727272727261 4.2499999999999994

Fitted data

Example 3:

Use `numpy.polyval` to generate x,y data of the fitted linear function.

```
In [47]: 1 x_new = np.array(sorted(x))           # x values, sorted monotonic
          2
          3 yfit1 = np.polyval(c1, x_new) # 1st degree polynomial
```

Try it yourself

Example 4:

Use `numpy.polyval` to generate x,y data of the fitted second degree polynomial function.

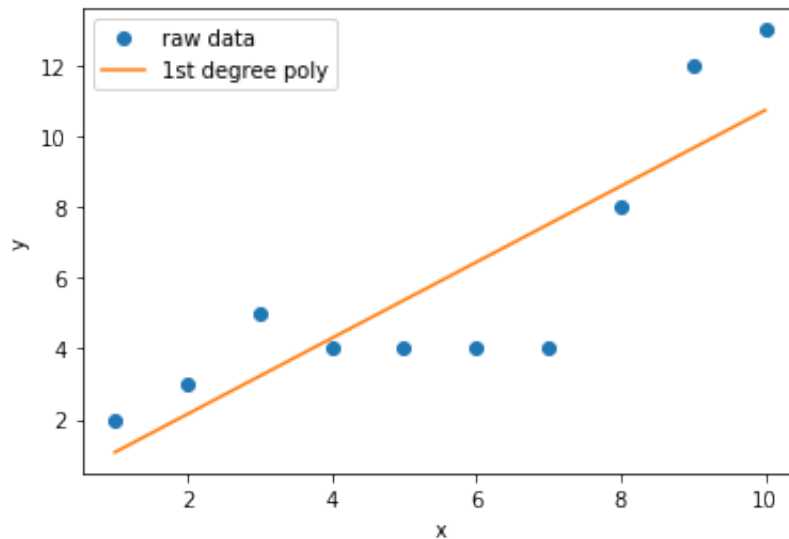
```
In [48]: 1 x_new = np.array(sorted(x))           # x values, sorted monotonic
          2
          3 yfit2 = np.polyval(c2, x_new) # 2nd degree polynomial
          4
```

Plotting fitted data

Example 5: Plot the raw data as a scatter plot and fitted linear function as a line graph ont eh same figure.

```
In [49]: 1 # plot data
2 plt.plot(x, y, 'o', label='raw data') # raw data
3 plt.plot(x_new, yfit1, label='1st degree poly'); # fitted 1st
4
5 plt.legend()
6
7 plt.xlabel('x')
8 plt.ylabel('y')
9
```

Out[49]: Text(0, 0.5, 'y')



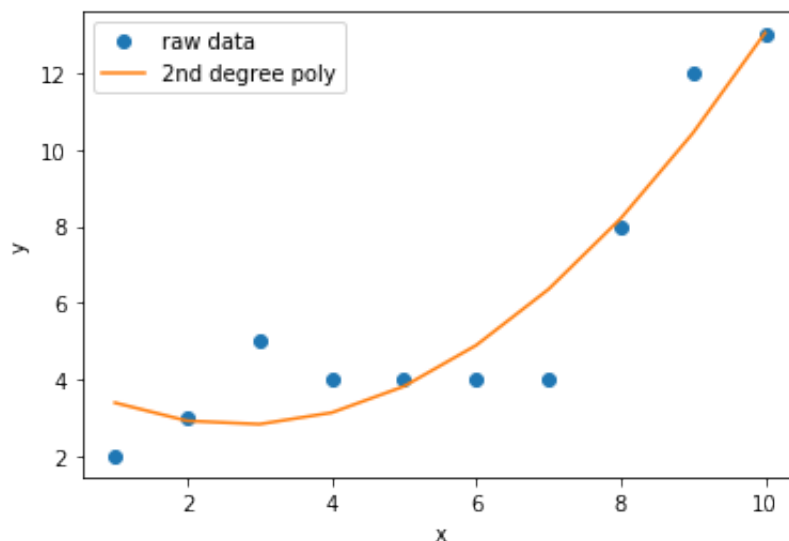
Try it yourself

Example 6:

Plot the raw data as a scatter plot and second degree polynomial function as a line graph on the same figure.

```
In [50]: 1 # plot data
2 plt.plot(x, y, 'o', label='raw data') # raw data
3 #plt.plot(x_new, yfit1, label='1st degree poly'); # fitted 1st
4 plt.plot(x_new, yfit2, label='2nd degree poly'); # fitted 2nd
5
6 plt.legend()
7
8 # label the axes
9 plt.xlabel('x')
10 plt.ylabel('y')
11
```

Out[50]: Text(0, 0.5, 'y')



Example 7

Fit the function $y = ae^{bx}$ which we defined earlier as `exponential` and find the RMSE:

```
In [51]: 1 from scipy.optimize import curve_fit
2
3 def RMSE(x, y, yfit):
4     "Returns the RMSE of a y data fitted to x-y raw data"
5     # error
6     e = (yfit - y)
7
8     # RMSE
9     return np.sqrt(np.sum(e**2) / len(x))
10
11 def exponential(x, a, b): # input arguments are independent var
12     y = a * np.exp(b*x)
13     return y
```

```
In [52]: 1 c, cov = curve_fit(exponential, x, y) # constants of fitted fu
          2
          3 yfit = exponential(x, *c)          # no need to sort x mono
          4
          5 rmse = RMSE(x,y,yfit)              # goodness of fit
          6
          7 print(f'RMSE = {rmse}')
          8
```

RMSE = 1.3338248760975626

In []:

1

In []:

1