Curve Fitting using Python

Objective:

1. To fit a linear and cubic polynomial for the Cp data.

Explain about:

1. How to make a curve fit perfectly?
2. How to get the best fit? Note: Best fit and perfect fit are two completely different phenomena.
3. What could be done to improve the cubic fit?

Questions:

What does popt and pcov mean ?

What does np.array(temperature) do?

What does * in *popt mean?

Given Data:

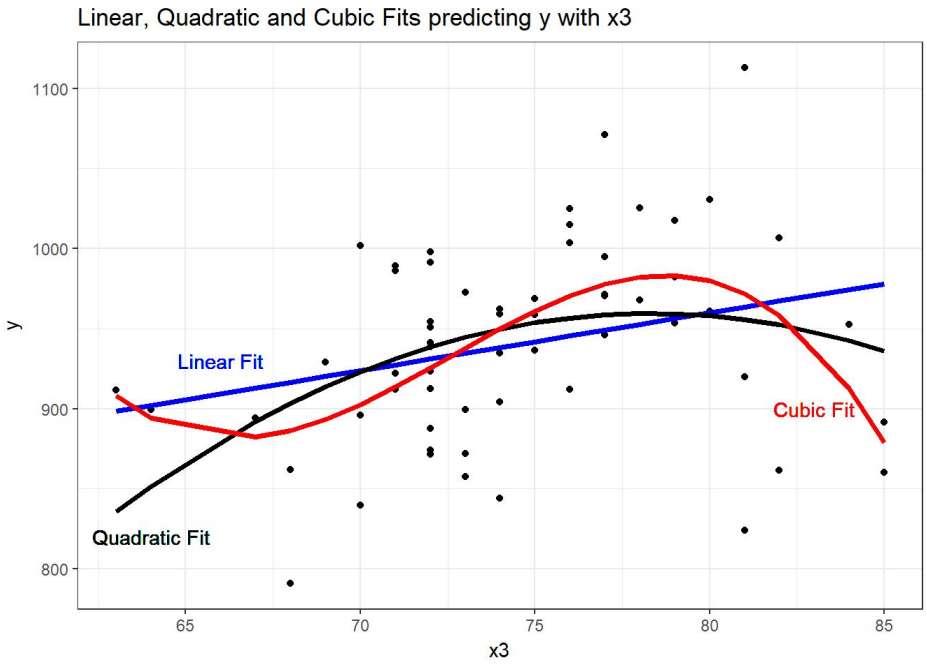
A set of Cp values against the temperature.

Procedure:

- Curve fitting is the process of constructing a curve that has the best fit to a series of data points provided.
- It can involve either interpolation, where an exact fit to the data is required, or smoothing, in which a "smooth" function is created that approximately fits the data.
- Here, Temperature and Cp data is loaded from a file.
- The coefficients for the linear and cubic equations are computed using polyfit.
  It returns the coefficients for a polynomial p(x) of degree n that is a best fit (in a least-squares sense) for the data in y.
  For the detailed comparison I have taken 4 polynomials
  cp=a*T+b for linear
  cp = a*T^2+b*T+c for quadratic
  cp=a*T^3+b*T^2+c*T+d for cubic
  cp=a*T^4+b*T^3+c*T^2+d*T+e for fourth order equation

- The value of Cp is predicted using polyval which is used for evaluating a polynomial at a specified value

Goodness of Fit:



As seen in the above example figure, goodness of fit increases with higher order polynomials.
Goodness of fit is a criteria to measure how good the data fitting is when compared one with another.

It can be assessed by four criteria. They are;

if $y = f(x)$ is the fit curve then the SSR is defined as follows,

$$SSR = \sum_{i=1}^{n} (f(x(i)) - Mean)^2$$

Similarly SSE is defined as follows,

$$SSE = \sum_{i=1}^{n} (Y(i) - f(x(i)))^2$$

Now the term SST is given by the summation of SSR and SSE

$$SST = SSR + SSE$$

From here, we can calculate the value of R^2,

$$R^2 = \frac{SSR}{SST}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (Y(i) - f(x(i)))^2}{n}} = \sqrt{\frac{SSE}{n}}$$

where,

n is the total number of datapoints available.

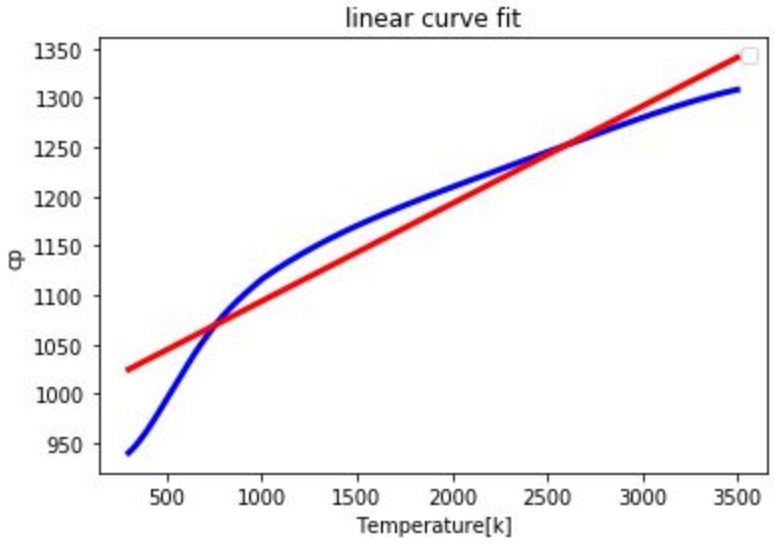
Program:

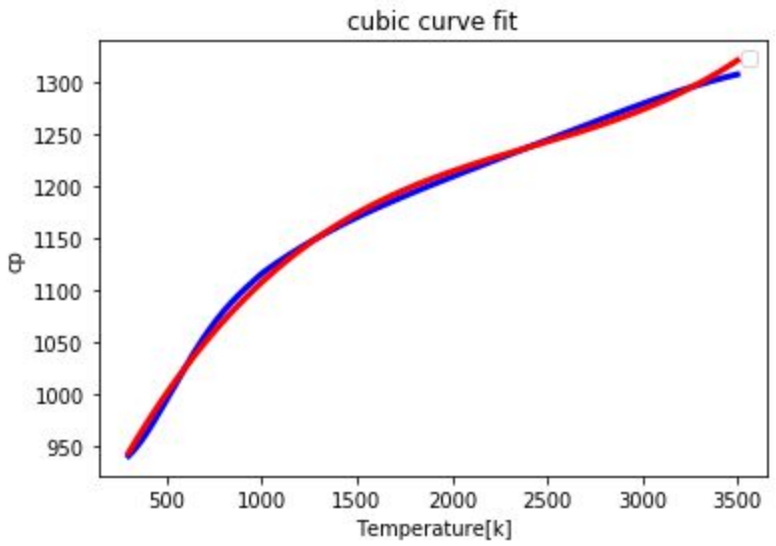
```
import math
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit


#defining the function for linear and cubic approximations
def linear(t,a,b,c):
    return a*t+b


def cubic(t,a,b,c,d):
    return a*pow(t,3)+b*pow(t,2)+c*t+d

#defining the function to read the file data and split it as per the
convenience
def read_file():
    temperature=[]
    cp=[]
    for line in open('data','r'):
        values=line.split(',')
        temperature.append(float(values[0]))
        cp.append(float(values[1]))
    return [temperature,cp]

#reading the file
```

```
temperature,cp=read_file()

#linear fitting
popt,pcov=curve_fit(linear,temperature,cp)
fit_cp_l=linear(np.array(temperature),*popt)

#cubic fitting
popt,pcov=curve_fit(cubic,temperature,cp)
fit_cp_c=cubic(np.array(temperature),*popt)

#plotting linear curve
plt.plot(temperature,cp,color="blue",linewidth=3)
plt.plot(temperature,fit_cp_l,color="red",linewidth=3)
plt.legend('Actual fit','Curve fit')
plt.xlabel('Temperature[k]')
plt.ylabel('cp')
plt.title('linear curve fit')
plt.show()

#plotting cubic curve
plt.plot(temperature,cp,color="blue",linewidth=3)
plt.plot(temperature,fit_cp_c,color="red",linewidth=3)
plt.legend('Actual fit','Curve fit')
plt.xlabel('Temperature[k]')
plt.ylabel('cp')
plt.title('cubic curve fit')
plt.show()

''' checking goodness of fit'''

addition=np.sum(cp)
size=np.size(cp)
mean=addition/size


'''linear fitting'''
lin_SSE=0
lin_SSR=0
for i in range(size):
    lin_error=abs((np.sum(cp[i]-fit_cp_l[i])))
    lin_SSE=lin_SSE+pow(lin_error,2)
    lin_SSR=lin_SSR+np.sum(pow((fit_cp_l[i]-mean),2))
```

```
lin_SST=lin_SSE+lin_SSR
print('linear_SST:',lin_SST)
lin_R=lin_SSR/lin_SST;
print('R of linear:',lin_R)
lin_RMSE=pow(lin_SSE,0.5)
print('RMSE of linear:',lin_RMSE)

'''cubic fitting'''
cubic_SSE=0
cubic_SSR=0
for j in range(size):
    cubic_error=abs((np.sum(cp[i]-fit_cp_c[i])))
    cubic_SSE=cubic_SSE+pow(cubic_error,2)
    cubic_SSR=cubic_SSR+np.sum(pow((fit_cp_c[i]-mean),2))

cubic_SST=cubic_SSE+cubic_SSR
print('cubic_SST:',cubic_SST)
cubic_R=cubic_SSR/cubic_SST;
print('R of cubic:',cubic_R)
cubic_RMSE=pow(cubic_SSE,0.5)
print('RMSE of cubic:',cubic_RMSE)
```

Results:





Inference:

It is clear that the splitwise curve fit method is more accurate than the normal fit. From the plots, if observed deeply, it is clear that the higher order curves have good fit compared to linear and quadratic.

1. How to make a curve fit perfectly?

Perfect fit is the one which fits the actual data points exactly and the error is almost equal to zero. If the R-square is equal to 1, it is the perfect fit.

2. How to get the best fit?

Best fit is the fit where the data points fits very close to the actual data and the error is minimum. If the R-square value is greater than 0.95, it is the best fit.

3.What could be done to improve the cubic fit?

In order to improve the cubic fit, higher order fit such as fourth order polynomials can be used. Using the splitwise method is also another solution to improve the cubic fit.

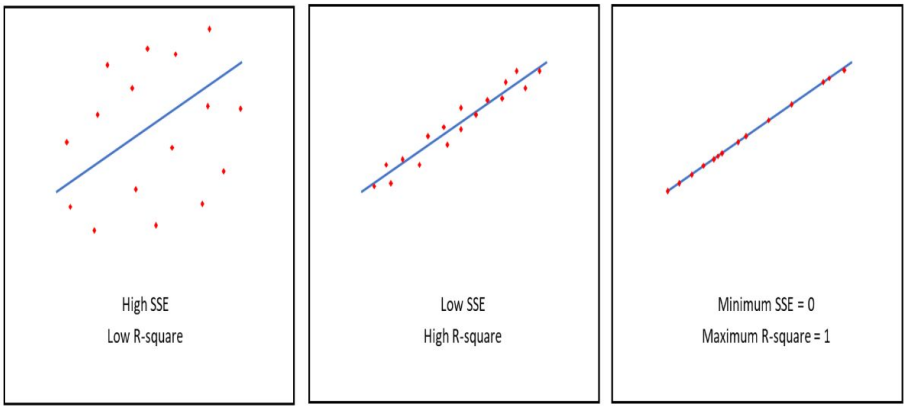
Goodness of Fit Results:

```
linear_SST: 28793673.249584787
R of linear: 0.9248776260524184
RMSE of linear: 1470.730801057686
cubic_SST: 62243756.575895965
R of cubic: 0.9907133759605076
RMSE of cubic: 760.2857134827851
```

Inference:

- Sum of Squares Total SST= Sum of Squares of Regression + Sum of Squares of Errors and R^2=SSR/SST. Generally, R^2 value between 0.95 to 1 is considered to be the best fit.
- In our results, R of linear is 0.92 whereas R of cubic is 0.99 which proves higher order polynomial to have the better fit.

All other outputs can be interpreted using the below diagram,



Questions:

1.What does popt and pcov mean ?

- popt-Optimal values for the parameters so that the sum of the squared residuals of f(xdata, *popt) - ydata is minimized.
- pcov-The estimated covariance of popt.

2. What does np.array(temperature) do?

List out the values of Temperature, stored in an array.

3. What does * in *popt mean?

It refers to the values stored in popt