Tobias Neckel Max-Planck, May 2019

Python Tutorial 6: Scientific Computing with Python

1) Polynomial interpolation

After some hours in the lab, you manage to obtain the following measurements:

x	1	2	3	4	5	6	7
y	1	2	1.5	2	3	2.5	4

Your goal is to try to fit these data to a polynomial function of the form

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6.$$

In other words, you want to find the coefficients $a_0, \ldots, a_6$ for which $p(x_i) = y_i$, where (x_i, y_i) are your measurements. The theory of quantum chromodynamics predicts that the values of the coefficients are

$$a_0 = -3., \quad a_1 = 4.033, \quad a_2 = 1.968,$$
$$a_3 = -2.885, \quad a_4 = 1.024, \quad a_5 = -0.147, \quad a_6 = 0.00763$$

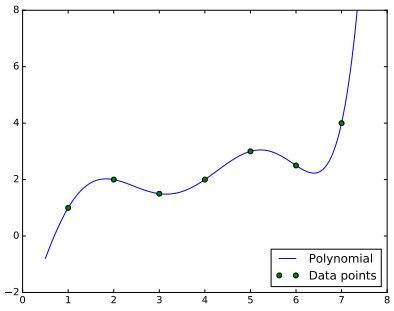

Figure 1: Polynomial interpolation of data points.

1. Create two NumPy Arrays `x` and `y` to store the measured data.
2. Plot the values using green circles.
3. The coefficients a_i can be found by solving the following system of linear equations $Ax = b$:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Create a matrix A and a vector b with the values from the equation.

4. Solve the system of equations using the function `scipy.linalg.solve(A,b)`.

5. Try to reproduce Figure 1.

6. An alternative method to compute the interpolation coefficients is by using the function `scipy.polyfit()`. Look up the reference for this function and use it to compute the coefficients a_i.
   Hint: Pay attention to the order (or usage) of the coefficients in `scipy.polyfit()`.

2) Fitting arbitrary functions

On the course website you can find a file called `data.txt`. This file contains two columns of data: the first column is the time t_i and the second is a set of measurements y_i. We now want to find the coefficients $p^* = (p_0, p_1, p_2, p_3) \in \mathbb{R}^4$ corresponding to the function

$$f(t) = p_0 \cos\left(\frac{2\pi}{p_1}t - p_2\right) + p_3 t,$$

that minimize the least squares error with respect to our data:

$$p^* = \arg\min_{p \in \mathbb{R}^4} \sum_{i=0}^{n} \left(f(t_i) - y_i\right)^2.$$

In other words, we want to find the parameters p of f that fit our data best, in the sense that they minimize the above expression (see Fig. 2). This is a common task in numerical mathematics, so there are many functions that do exactly this. Use the function `scipy.optimize.leastsq` to find the optimal parameters p to fit the data (look up the documentation). Plot your results using `matplotlib`.

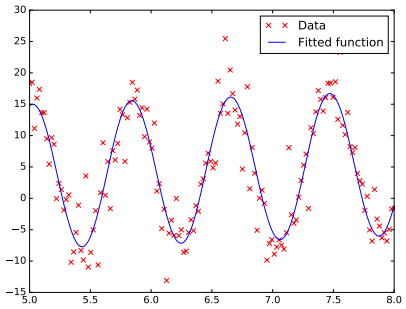

Figure 2: Fitting a sine function in the least squares sense.