IITM-ID2090 : Introduction to Scientific Computing
Final Assignment — Due Date : 31 July

Roll No: ME19B193 — Name: Shreyas Kulkarni

1. Consider a symmetrical natural shape such as what is shown in the figure in the assignment. Utilize the contrast difference between the shape (flower in this case) and the background (green leaves in this case) to determine the contour of the shape. Convert the contour to a set of points that represent the outer form of the shape. Fit a function $r(\theta)$ such as to unwrap the shape to a linear function of the angle in the polar form. You can utilize step functions or splines or a combination of any elementary mathematical functions to fit. Plot the function and show it side-by-side with the original image to confirm that the procedure is able to capture the shape correctly. Compare the size of the original image (bytes) with the data required to store the function (bytes) and comment on the compression ratio achieved. Comment also on the possibility of determining the symmetry of the image from this exercise. Eg., the above image has a 8 fold rotational symmetry. Test your code by using any other image of a symmetrical flower taken from the internet.

   Application: Image compression for the purpose of comparison of similar images is an important aspect of image recognition. If this were to be done in a video, the size and speed matter a lot given that large number of images that need to be processed in real time. Recognizing symmetry in an object is non-trivial if the angle of view is not along the axis of symmetry.

   Solution:
   Required files:

   - code.ipynb : Required Jupyter Notebook with all the required codes
   - flower.png : The given image for experimentation
   - flower net1.png : Test Image from the internet

   Methodology and Assumptions:

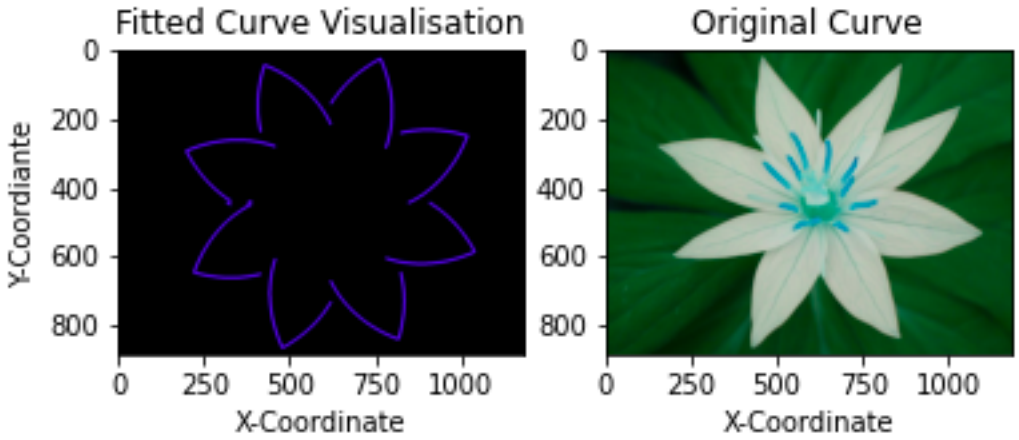
   - The Contour of the Image is determined by using opencv functions.
   - The centre is then found by considering 4 extreme points of the contour. A curve is fit between the topmost, bottommost, leftmost and rightmost points by using curve fit of scipy[2].

- A linear function is considered by using modulus to considering symmetry.

```
def test_func(theta,a,b,c, d):
    temp = theta*180/np.pi % a
    return b*np.abs(temp - c) - d
```

- The Curve is then fit again by using scipy's curve fit [2].
- To determine the symmetry of the flower, the peaks of the curve are counted.

Curve Fitting on the Given Image:



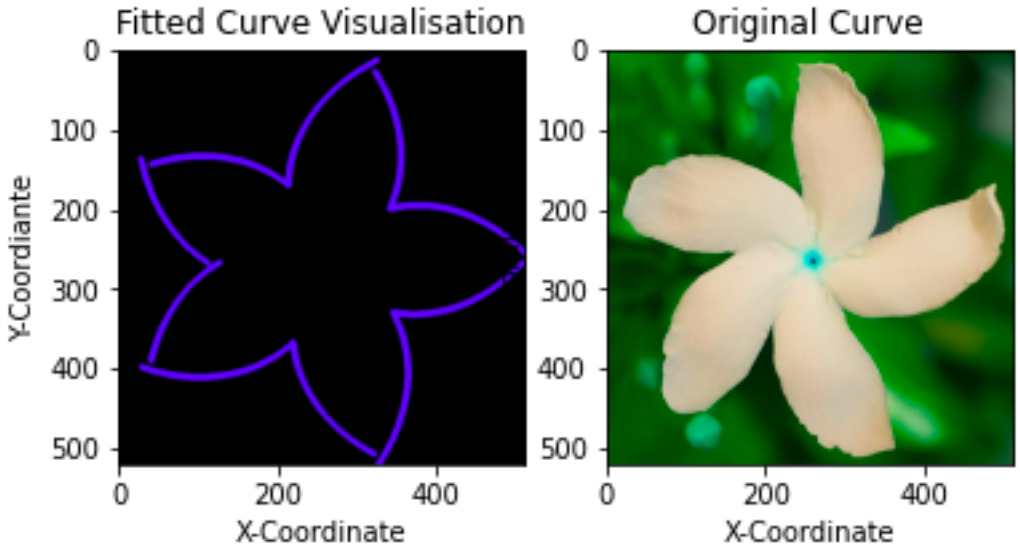
Compression Ratio on the Given Image:

$$\frac{\text{Size of the Original Image}}{\text{Size of the compressed Curve}} = 23069.5$$

A very high compression ratio is achieved. Clearly this a great technique to extract and save the curve.

Determining Symmetry:
Symmetry can be determined by the number of peaks in the curve. This can be found by find peaks function of scipy. The symmetry achieved in this is 8 fold. (Calculated and Validated in the code).

Curve Fitting on the Internet Image:
Source of Image [3]



This failed to store the shape of the curve much accurately. However it did a great job in extracting the symmetry and positions of the points. **Compression Ratio on the Internet Image:**

$$\frac{\text{Size of the Original Image}}{\text{Size of the compressed Curve}} = 5885.23$$

The difference is compression ratio is just because the internet image is smaller than that of the given image. The size of the compressed curve shall remain the same.

Determining Symmetry:
The symmetry achieved in this is 5 fold. (Calculated and Validated in the code).

2. Consider a function of a form such as $g(x) = A_1 + A_2 exp(-A_3 x)$ where A_1, A_2 and A_3 are constants. Assuming random values for these three constants, generate the values of the function g_i for a set of input values x_i that lie in the interval between 0 and 5. Make a scatter plot of (x_i, g_i) which can be taken as the experimental data. You can consider the number of values to be 50, to be representative.
   Add a random noise δ_i of amplitude 5% of the maximum value of this set g_i to each of the values. The random noise can be either positive or negative. The data thus generated will be $(x_i, g_i + \delta_i)$. Now, fit the same function $h(x) = B_1 + B_2 exp(-B_3 x)$ as a model using this input data. Superpose the original function and the fitted function to confirm that the fitting process was reasonable. Comment on the difference between the initial values of A_1, A_2 and A_3 with those of B_1, B_2 and B_3 obtained from the fit. Change the amplitude of the noise from 5% to 7% and then 10% to see if it has any influence on the differences $(A_1 - B_1), (A_2 - B_2)$ and $(A_3 - B_3)$. Comment on the trend.

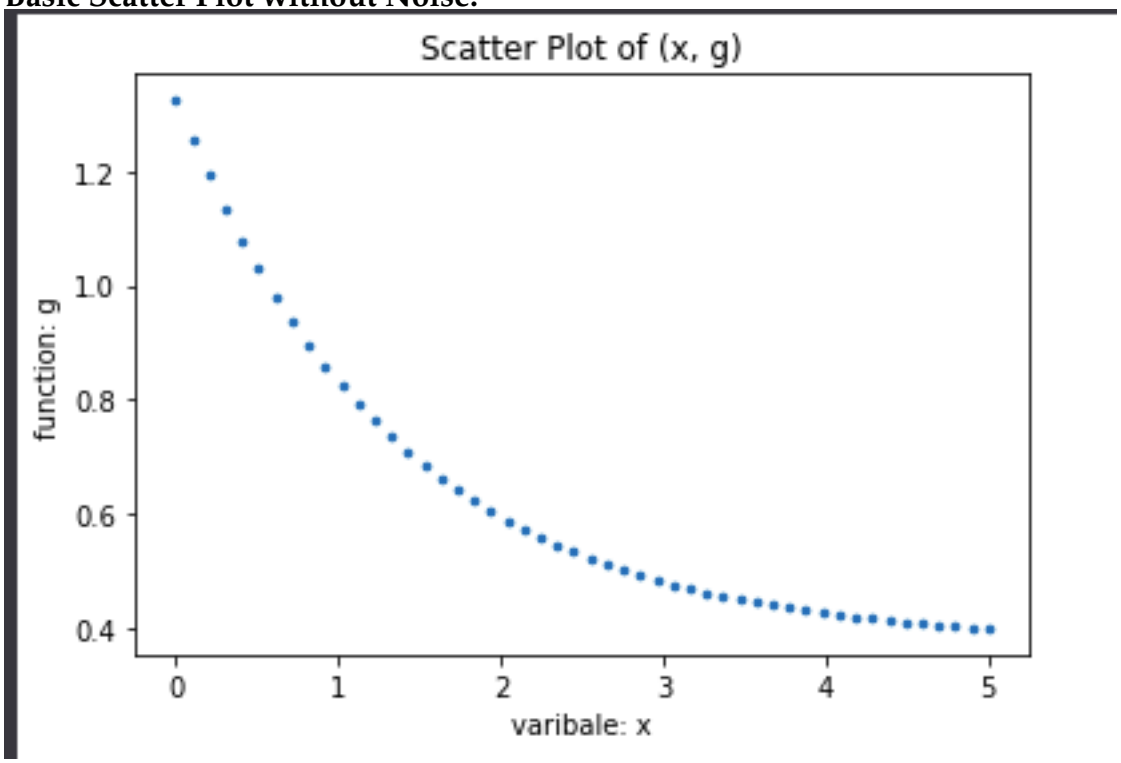
Application: Experimental data usually has a noise. One should fit experimental data only to those mathematical forms / models that have some physical significance with the phenomena associated with those experiments. Error in the parameters of the models can be reduced if more information is available about the physical phenomena.

Solution:
Required Code: code.ipynb (Jupyter Notebook)

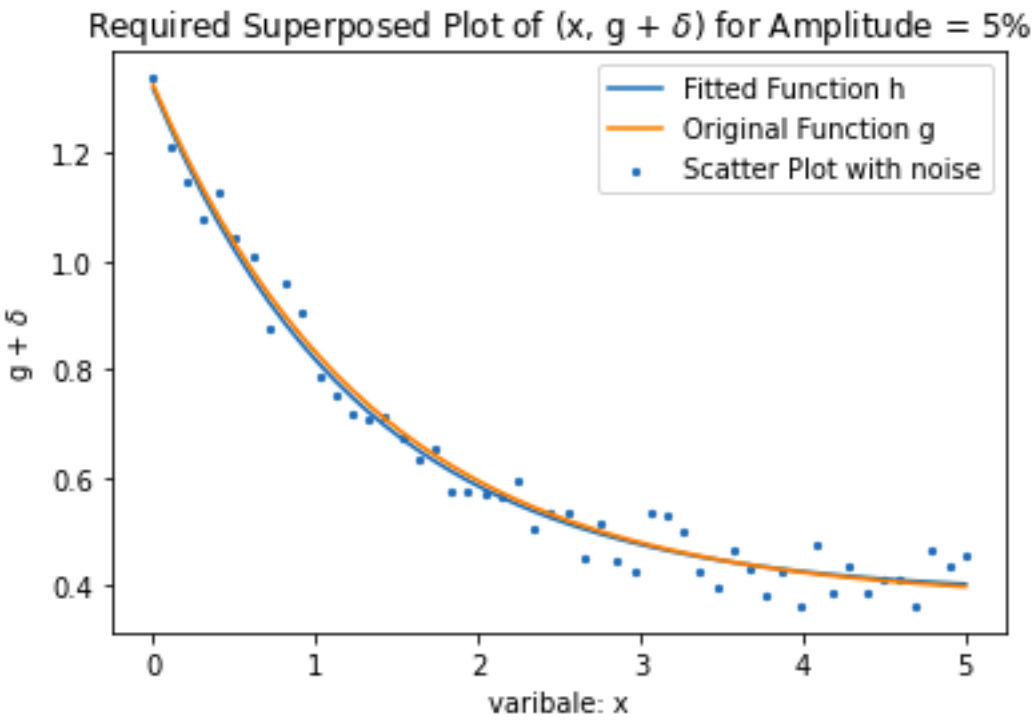
Assumptions:

- The random values for the 3 constants A_1, A_2 and A_3 is chosen uniformly between 0 and 1 [1]
- The random noise is also decided by sampling uniformly between 0 and $+-$ the amplitude [1]

Basic Scatter Plot without Noise:



Fitting with Noise Amplitude 5%:



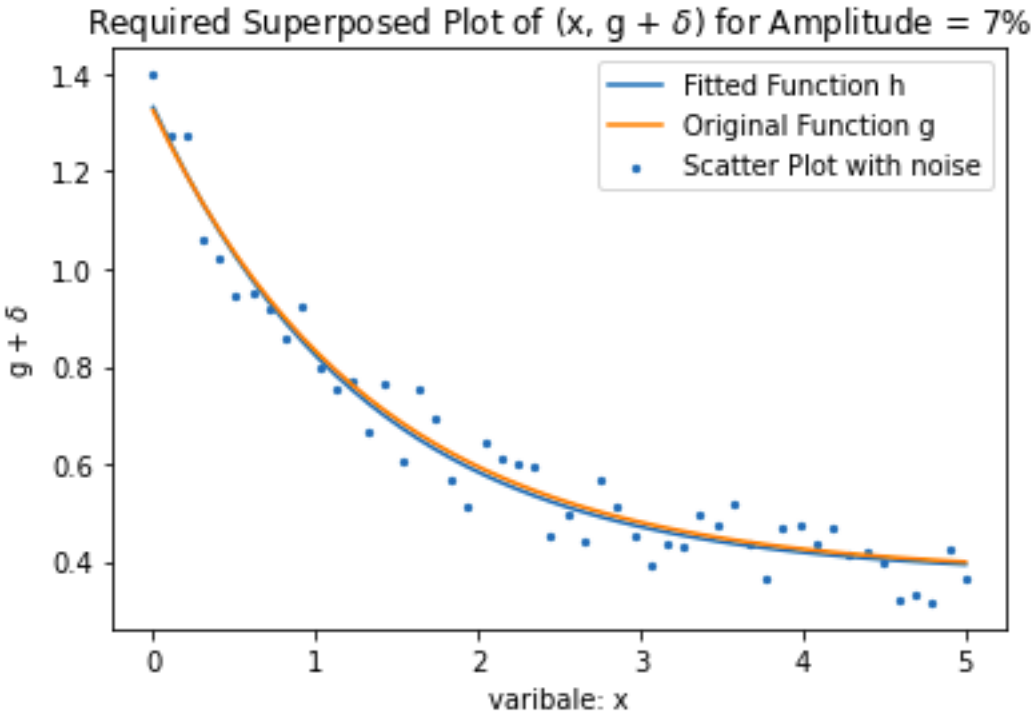
This confirms that the fitting process was reasonable.

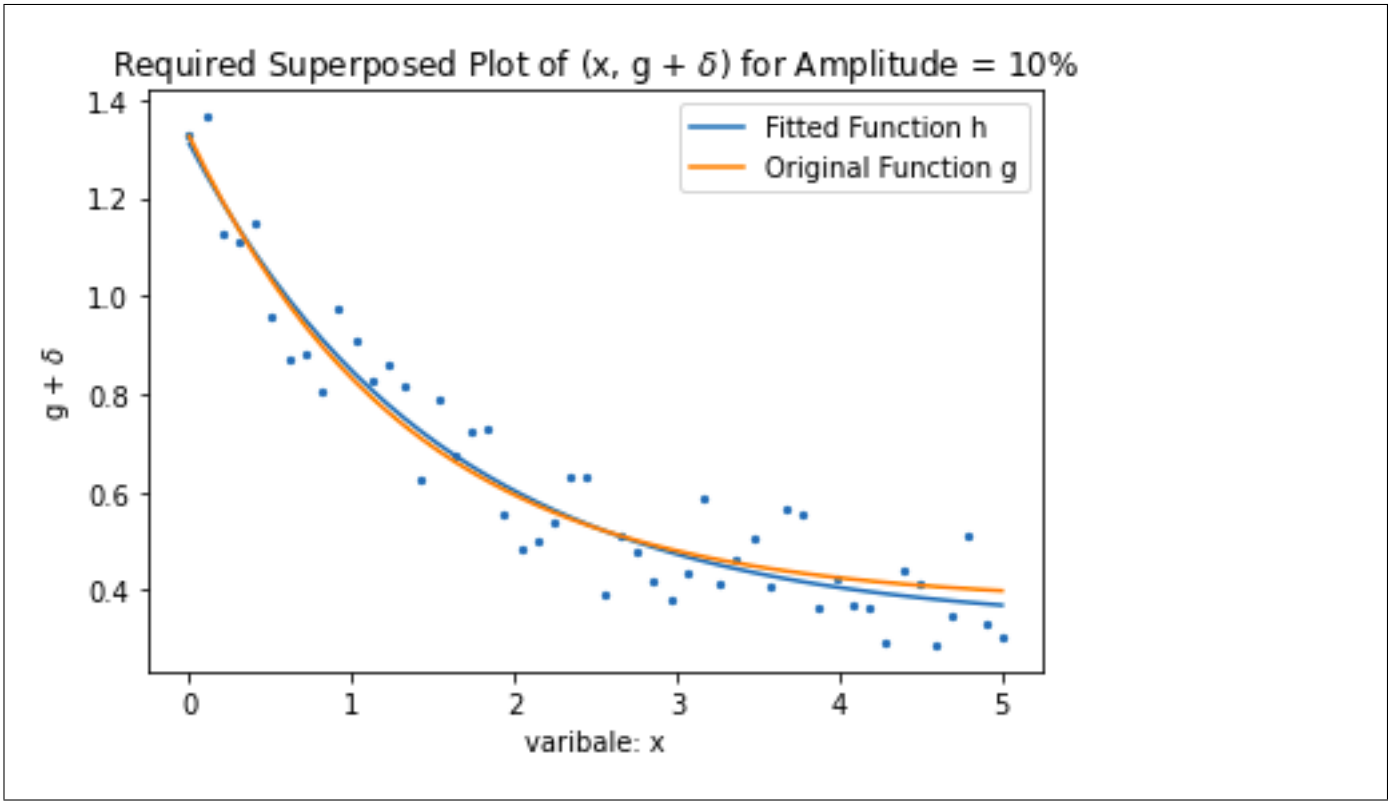
Comment on difference between Initial and fitted values of the parameters for 5% Amplitude:
$[0.00899819, -0.01490829, 0.03597984]$
At $x = 0$, $h = B_1 + B_2$ which means that $B_1 + B_2$ depends on h closer to 0.
As $x \to \infty$, $h = B_1$ which means B_1 depends on h far away from 0.
At $x = 0$, $\frac{dh}{dx} = B_2 * B_3$ which means that $B_2 * B_3$ depends on change in h closer to 0.
From above, we can see confirm that the fitted curve h is has some major differences at the middle while not much change can be seen near 5 and 0. Therefore the slope is impacted more which depends on B_3 majorly and it has the highest percentage of change (4.9%).
The difference between the parameters depend on the randomness and the fitting that has been done.
The percentage of difference between these parameters does not exceed 5% which tells us that the noise hasn't disturbed the curve fitting much.

Comment on trend of difference in parameters%:
Required Table:
This table represents the percentage difference in the parameters.

	A1-B1	A2-B2	A3-B3
5%	2.402463	1.568115	4.915320
7%	0.671552	0.938718	3.281536
10%	12.058134	3.201828	12.705088

Usual intuition says that the percentage of difference in parameters should increase with increase in the amplitude of the noise. However, it doesn't seem to happen so here.
One major reason is that the difference in paramaters depends on the distribution of the noise rather than the amplitude of noise itself. For instance, if two close points with one point having high positive amplitude and the other point having high negative amplitude would just cancel out to have a more closer to g curve. While if all the points have a small positive deflection then the curve would be shifted (major change). This can be seen in the graphs. The noise with Amplitude 7% have closeby points with negative and positive deflection. This is why they have small change in the parameters as compared to 5%. While amplitude with 10% just has very high deflections causing a major change in the parameters.
The other plots can be seen here:





References

[1] Numpy Docs

[2] Scipy Docs

[3] Wikpedia Image