EE1103: Numerical Methods

Programming Assignment # 5

Lakshmiram S, EE22B117
Collaborator: Hrushikesh Kant, EE22B108

January 9, 2023

Contents

1 Problem 1

It is known that the data tabulated below can be modeled by the following equation

$$y = \left(\frac{a + \sqrt{x}}{b\sqrt{x}} \right)^2 \tag{1}$$

Use a transformation to linearize this equation and then employ linear regression to determine the parameters a and b. Based on your analysis predict y at x = 1.6.

1.1 Approach

We first linearize the given function

$$\sqrt{y} = \frac{a}{b}\frac{1}{\sqrt{x}} + \frac{1}{b}$$

this can be represented in the form of a straight line as $y = a_0 + a_1 x$ where $y' = \sqrt{y}$, $a_1 = \frac{a}{b}$, $a_0 = \frac{1}{b}$ and $x' = \frac{1}{\sqrt{x}}$.
We can modify the data points according to the above transformation and treat the resulting line as the linear regressor.

1.2 Algorithm

The same as that in the course material.

1.3 Results

The required values of a_0 and a_1 obtained are 4.861448 and 2.440323 respectively.
The value of the function at x=1.6,predicted using regression is 3.597195.

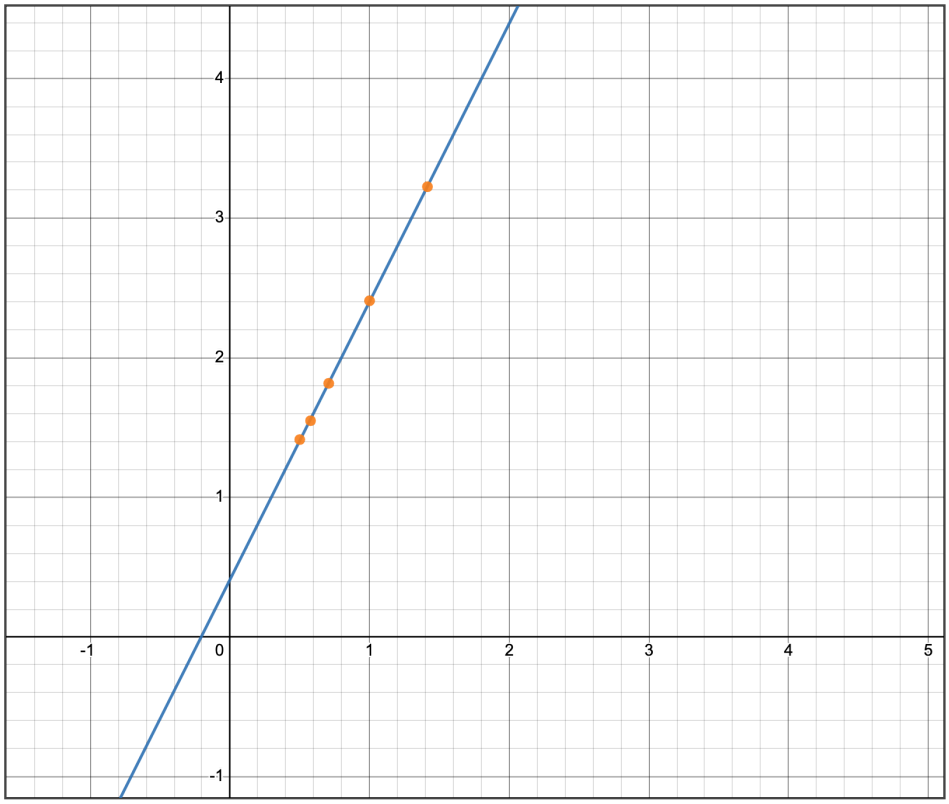

Figure 1: the linear regression line along with the data points.

1.4 Inferences

- Linearisation is one elegant way to convert seemingly complex functions into easy-to-handle straight lines.
- Here the regression line passes through all the data points. While this might not be the case with all regressions, we are still guaranteed that least squares regression will

yield a single "best fit" curve for every case.

1.5 Code

I am not including the code here as it is being submitted separately.

1.6 Contributions

I worked on this question independently.

2 Problem 2

Using multiple linear regression , fit a curve for the given set of data points.Also compute the standard error of the estimate and the correlation coefficient.

2.1 Approach

This is a pretty straightforward question. The theory to solve this is well established and is just linear regression extended to 2 independent variables .(refer Chapra)

2.2 Results

The output we get is:
y = 14.460876 + 9.025223 x1 + -5.704361 x2
The Standard error of estimate is : 0.888787. Correlation coefficient is: 0.997759

2.3 Inferences

- The idea of multiple regression is to extend the concept of linear regression to more than one independent variable. For 2 independent parameters,(x_1 and x_2) the regressor is a plane (in contrast to a line in the case of one independent variable). For more than 2 variables, the regressor can't be visualised.

2.4 Contributions

This code was contributed by my collaborator .

3 Problem 3

Fit the best 3,4,5,6 degree polynomials on Shashi tharoor' hairline.

3.1 Approach

The general n^{th} degree polynomial has $n+1$ independently variable parameters and so we need $n+1$ coordinates that pass through the curve to uniquely determine that curve.
There are quite a few ways to go about doing this:

1. Assume a general polynomial of the form $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$. Now write down the constraint equations-$f(x_i) = y_i$ for all the $n+1$ data points.We can solve for $n+1$ equations in $n+1$ unknowns by methods like LU decomposition.

2. The newton interpolation polynomial expression provides for a nice recursive computer implementation of the same technique as above.
   Here we assume $f(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \cdots + b_n(x - x_0)(x - x_1)\ldots(x - x_{n-1})$ where b_i's can be solved by a procedure (refer Chapra).

3. A third method is provided by Lagrange polynomial where the whole polynomial is fixed ie., the coefficients are already determined(in terms of the data points).

3.2 Results

3rd degree curve:
points taken:
equation obtained:

Table 1: randomly chosen coordinates

6.3	10.1
8.9	15.8
13.6	16.35
15.13	13.33

$f(x) = 10.100000 + 2.192308(x - 6.300000) + -0.284286(x - 6.300000)(x - 8.900000) + -0.005813(x - 6.300000)(x - 8.900000)(x - 13.600000)$

curve superimposed on the image:

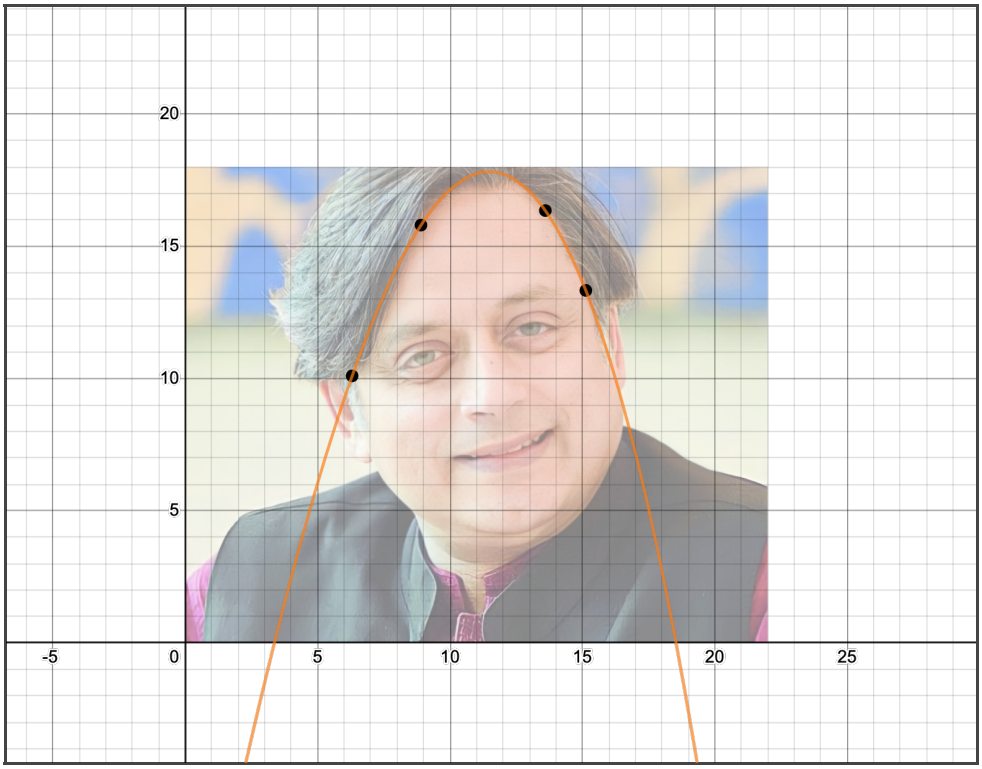

Figure 2: degree 3 interpolation

4rd degree curve:
curve superimposed on the image:

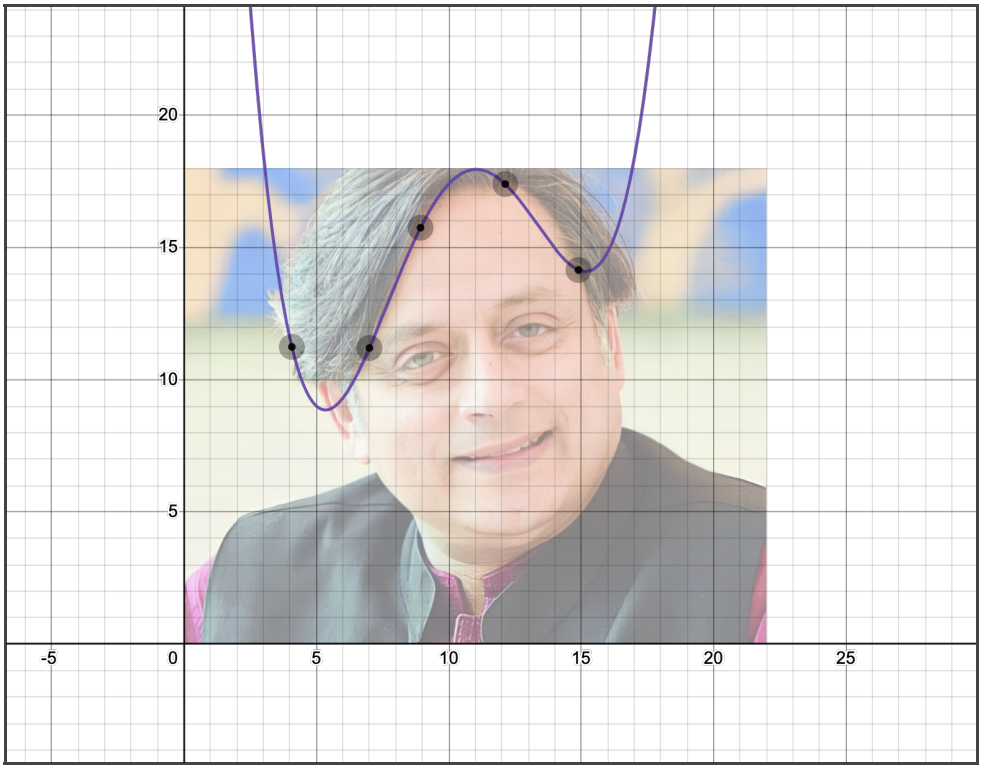

Figure 3: degree 4 interpolation

5rd degree curve:
curve superimposed on the image:

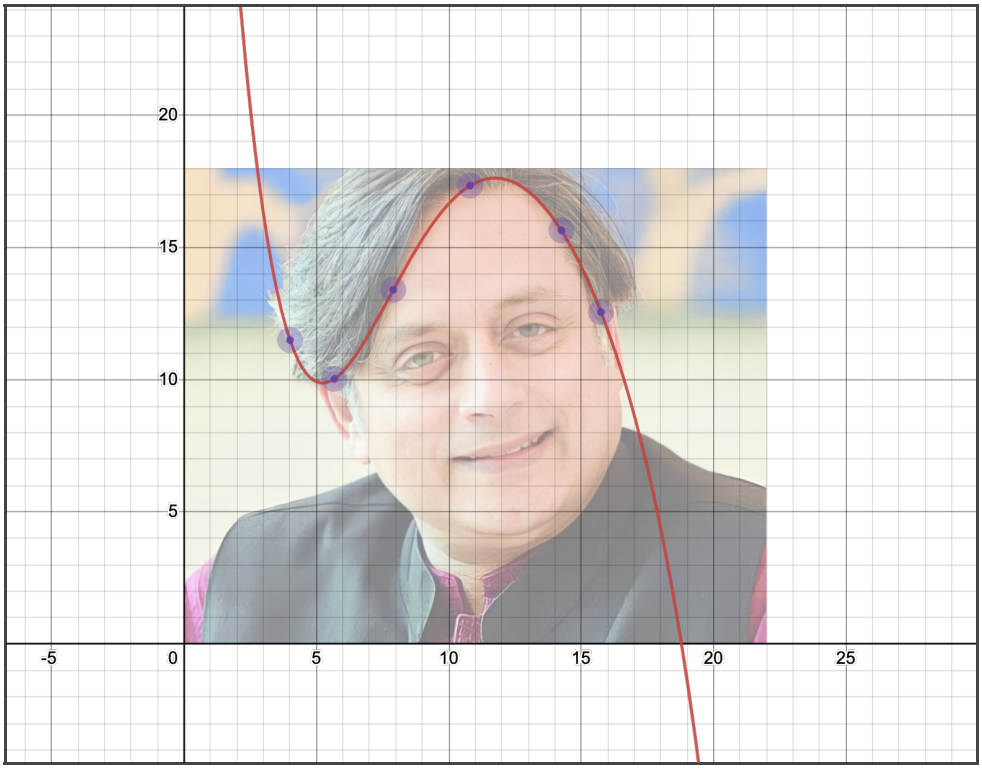

Figure 4: degree 5 interpolation

6rd degree curve:
curve superimposed on the image:

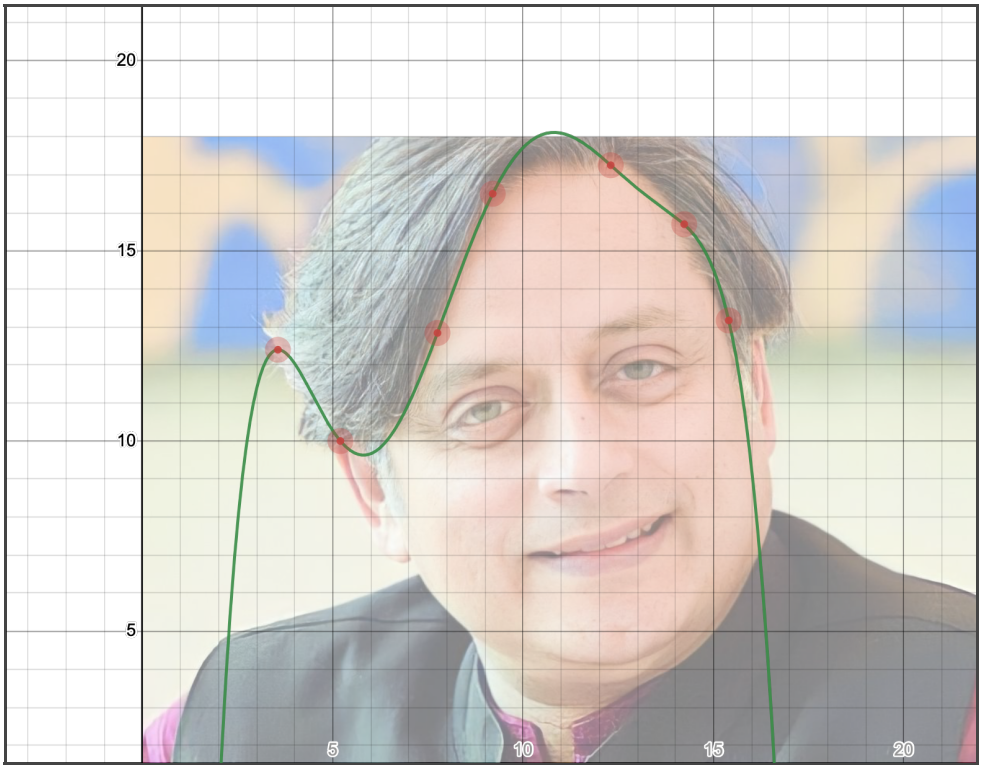

Figure 5: degree 6 interpolation.

3.3 Inferences

There weren't any important deductions in this question as such. It was just a matter of implementing the suggested theory as computer code.

3.4 Contributions

I worked on this question independently.

4 Problem 4

Fix 7 salient points on Shashi Tharoor's hairline and fit the best quadratic and cubic splines in each interval. Juxtapose your splines with the image

4.1 Approach

For quadratic splines:
We have $n+1$ points and so n intervals , each corresponding to n quadratics. There are several constraints to determine these curves uniquely (refer Chapra). We can deal with this problem in a couple of ways: either assume the general expression of a polynomial as $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$, use the constraints and solve some $m \times m$ matrix system.Or, we could employ the general expression for a polynomial (here a quadratic): $f(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$. This form is advantageous over the others as this covers up a few constraints in its very construction. This way a_0 and a_1 are pre-determined.All we have to do is write code for determining the a_2's of all the curves, accounting for n variables.

For cubic splines:
There's a pretty neat method mentioned in Chapra to deal with cubic splines, which I have implemented in my code.

4.2 Results

The image so obtained after superimposing the 6 quadratic splines after choosing 7 random points on the person's hairline is :

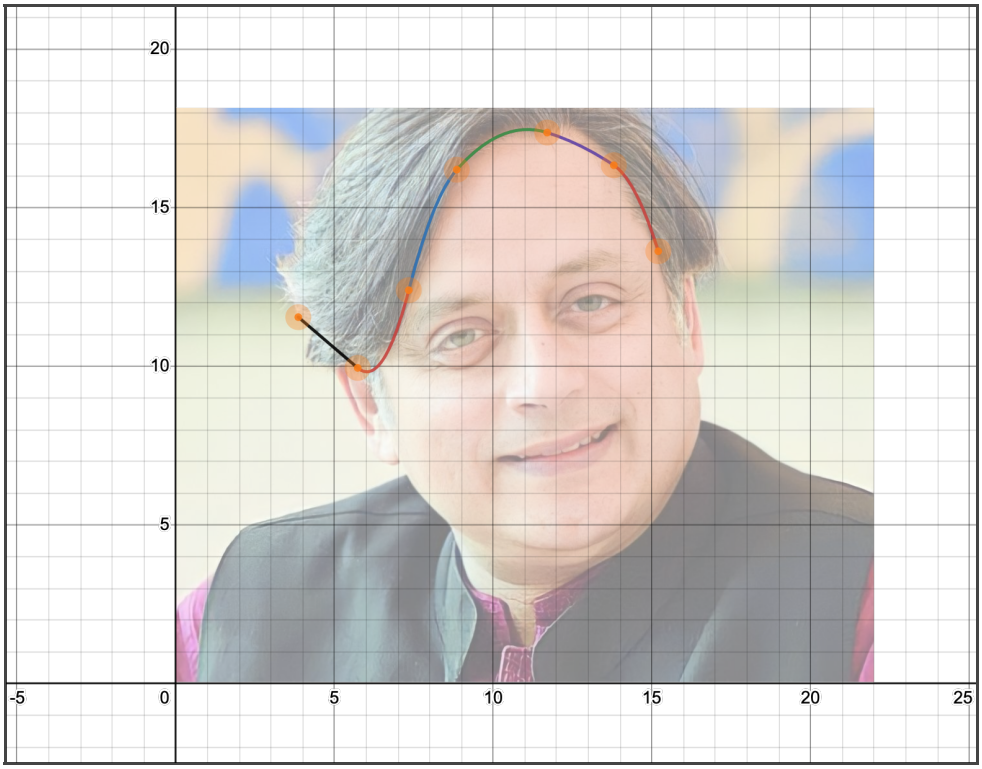

Figure 6: quadratic splines

The image so obtained after superimposing the 6 cubic splines after choosing 7 random points on the person's hairline is :

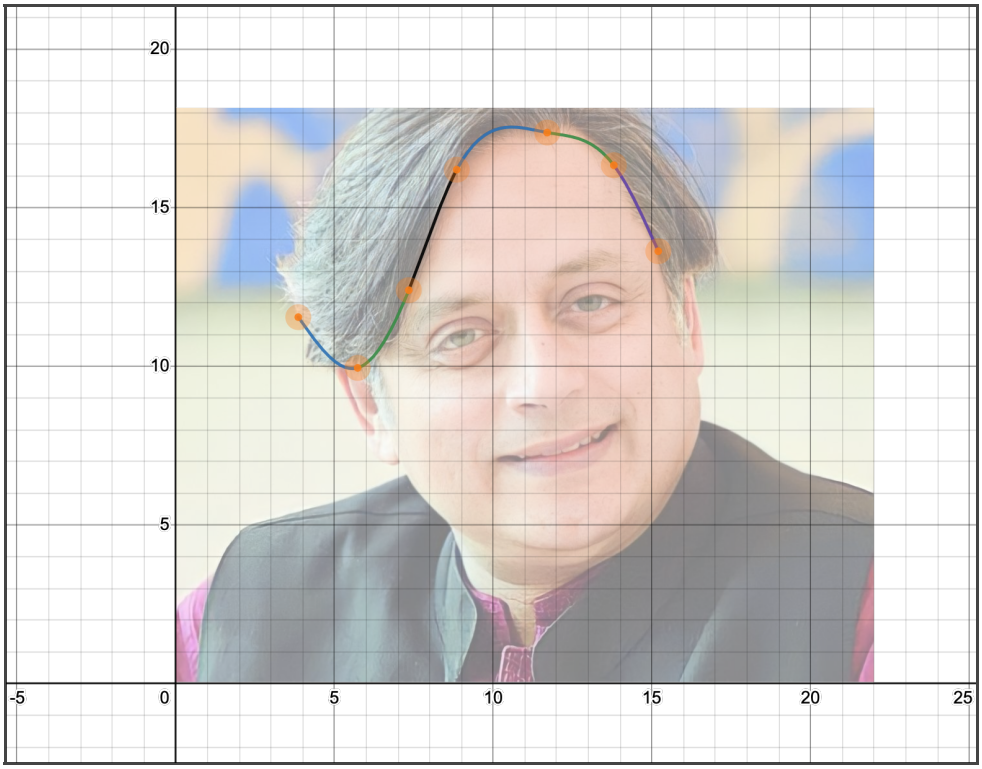

Figure 7: cubic splines

4.3 Inferences

Using the Newton way of expressing a polynomial comes quite handy here.Speaking of handy, the Lagrange way of representing the general polynomial is even better, as all the coefficients are already determined (in terms of the data points),which means there is no equation-solving involved, and the computer implementation of the same is also quite straightforward.

4.4 Contributions

I worked on this question independently.