

**CE7453 Numerical Algorithms**

Assignment 1: Cubic B-Spline Interpolation

He Wei

G1802216K

School of Computer Science and Engineering

\

1. **The Linear system required in item2)**

With the input data:

[0 1], [1 0], [1 1], [0 1], [0 2]

The matrix N in the constructed linear system is as follows:

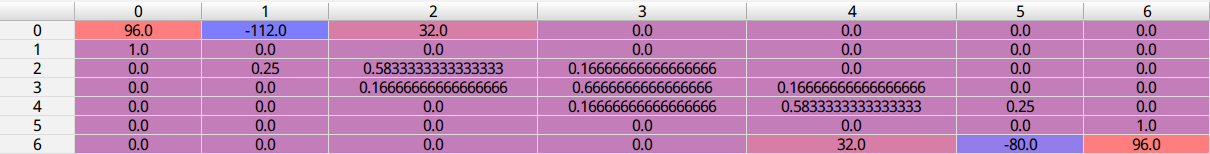


Figure 1. Matrix N

Using the cubic B-spline basis from the slides in module 5:

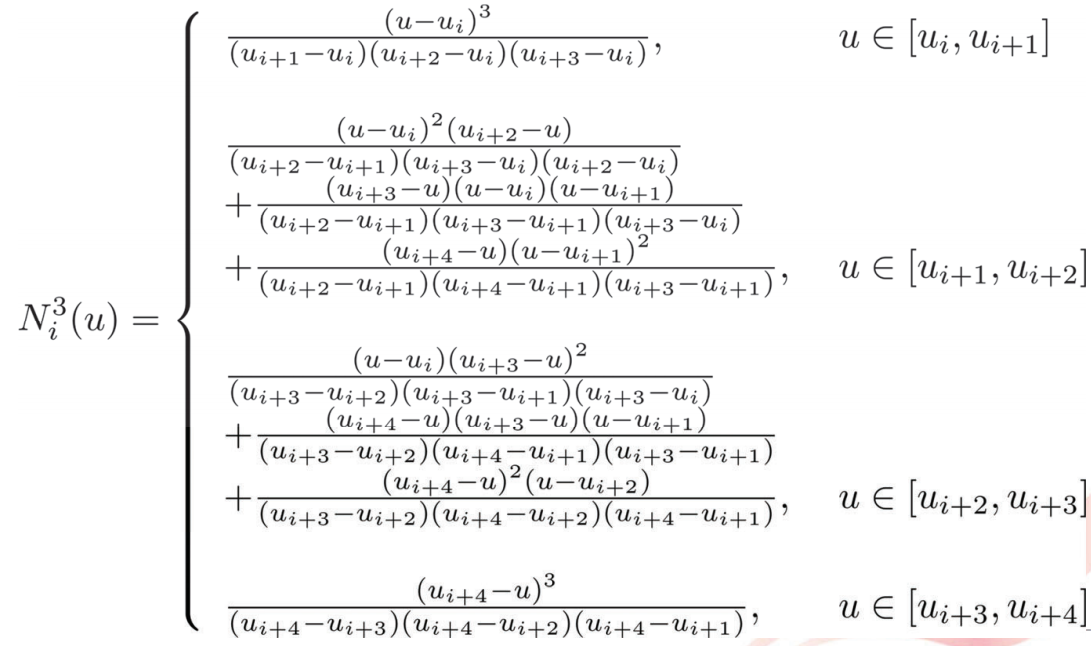


Figure 2. Cubic B-Spline Basis Functions

Hence, the linear system D=N\*P, where the D is the set of data points, N is a (n+3)\*(n+3) constraint matrix that contains n+1 data point fitting equations and 2 end points condition equations, and P is the set of control points. The linear system can be expressed into the following matrix form:



where for the item 2 is as follows:



Solving this system, we may use LU decompostion, the result of solved control points are as follows:



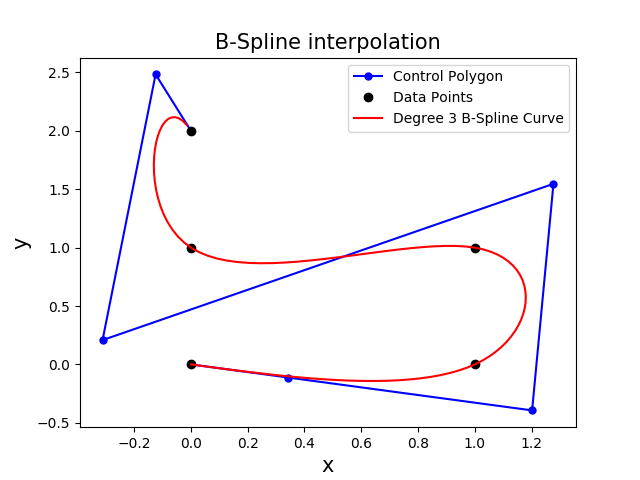


Figure 3. Visualization of the Curve in item2)

1. **The data of two examples and the output of the program**

**Example 1**:

Input Data Points(13 in total):

[-5 0], [-2 10], [-1 -13], [0 15], [1 13], [2 10], [5 0], [2 -10], [1 -13], [0 -5], [-1 -13]. [-2 -10], [-5 0]

**Example 2**:

Input Data Points(11 in total):

[2, 0], [10, 20], [15, 10], [20, 19], [22, -5], [37, 7], [50, -5], [30, -7], [20, -10], [-5, -15], [-20, -30]

**Solution for Example 1& Example 2:**

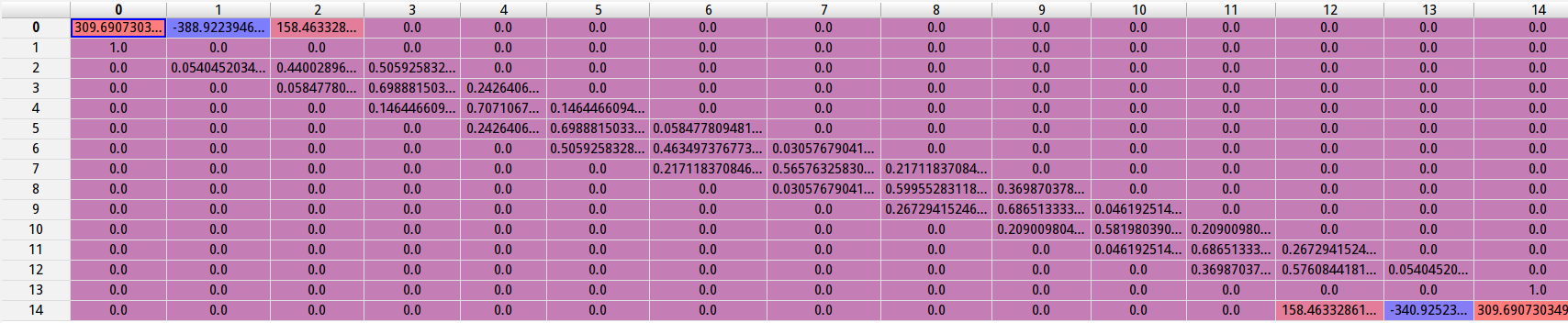
**Constructed Knots Vector for Example 1**(19 in total):

[0, 0, 0, 0, 0.139, 0.181, 0.211, 0.241, 0.283, 0.422, 0.562, 0.604, 0.711, 0.819, 0.861, 1, 1, 1, 1]

**Constructed Control Points for Example 1**(15 in total):

[-5.0 0.0], [-5.01 3.32], [-2.53 8.14],[-1.22 12.33], [0.0 16.11], [1.22 12.33], [2.54 8.17], [6.81 -0.7], [2.75 -6.34], [0.38 -16.69], [0.01 3.4], [-0.42 -16.69], [-2.66 -6.37], [-5.78 -2.96], [-5.0 0.0]

**Matrix N in for** **Example 1**(15\*15):



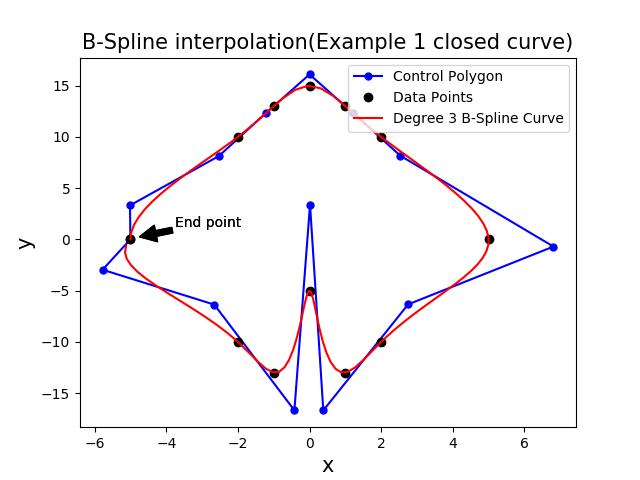


Figure 4. Interpolating Curve for Example 1

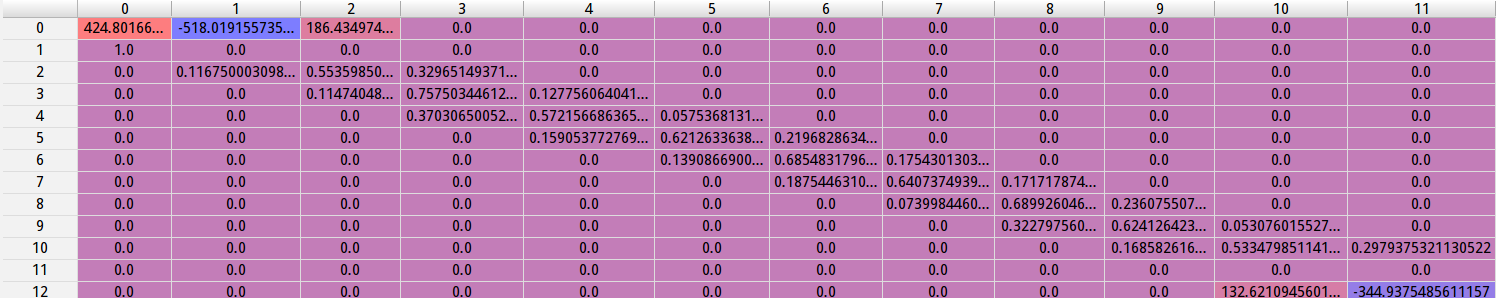
**Constructed Knots Vector for Example 2**(17 in total):

[0, 0, 0, 0, 0.119, 0.181, 0.237, 0.370, 0.476, 0.574, 0.685, 0.742, 0.883, 1, 1, 1, 1]

**Constructed Control Points for** **Example 2**(13 in total):

[2.0 0.0], [4.68 11.56], [8.45 32.12], [14.49 2.63], [23.89 33.82], [16.76 -23.05], [35.44 17.94], [59.14 -11.93], [31.8 -4.18], [15.61 -13.68], [-0.1 -2.07], [-25.44 -38.89], [-20.0 -30.0]

**Matrix N in for Example 2**(13\*13):



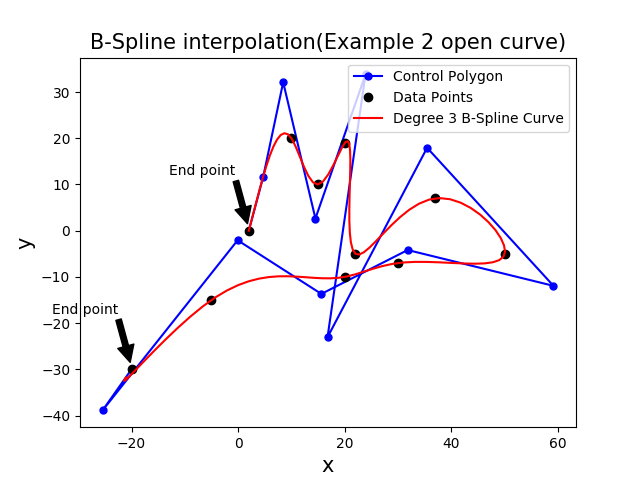


Figure 5. Interpolating Curve for Example 2

1. **Things to highlight:**

3.1 Parameterization

In here, the program is using the Chord Length Paramerization method, which implements parametric equation to represent the curve and the parameters are computed based on the spatial location of the data points.

For example, in Example 1, we can construct the knot vector based on Uniform parameterization or Chord Length parameterization:

**Constructed Knots Vector for Example 2 using Chord length parameterization:**

[0, 0, 0, 0, 0.119, 0.181, 0.237, 0.370, 0.476, 0.574, 0.685, 0.742, 0.883, 1, 1, 1, 1]

**Constructed Knots Vector for Example 2 using Uniform parameterization:**

[0, 0, 0, 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1, 1, 1]

The constructed control polygon of these two methods has slightly difference, but the interpolating curves still go through the same data points. (See Figure 6.)

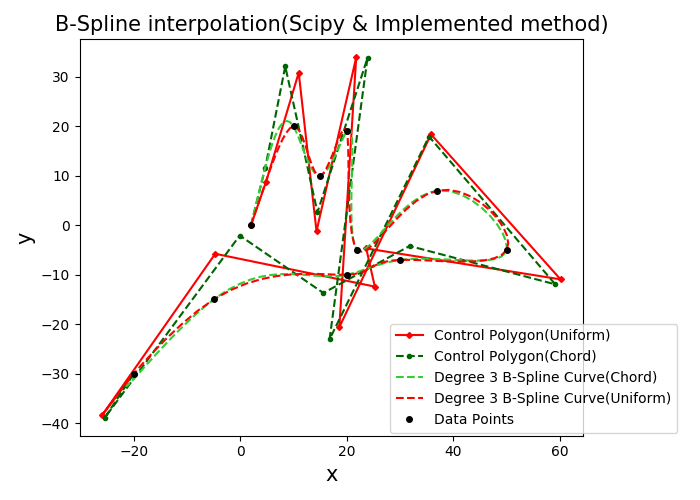


Figure 6. Comparison between two parameterization methods

3.2 Difference of the basis function between scipy package and the implemented ones

Due to the slight difference between the basis function, the implemented method differs from the scipy implementation. However, they all satisfy the requirement and interpolate between the given data points. (See Figutre 7.)

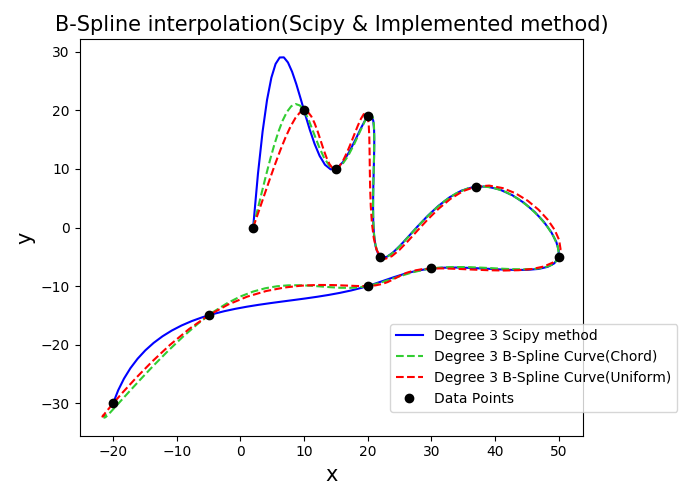


Figure 7. Scipy implementation & self-implemented methods

3.3 GUI software

To better visualization, the program for this assignment has been developed with a user interface. The software includes functions like plotting the interpolating curve, calculating the control points based on the data points, configure the coefficient matrix N, and plotting the data points scatter diagram or the control polygon diagram.

The initial GUI window are shown in the Figure 8, which includes the plotting canvas, the text browser and a list of buttons.

To plot the interpolating curve, the following instruction are needed:

* **Load the data.txt, which contains the data points coordinate (x, y):**

The default data.txt lists the data points that are in item 2): [0 1], [1 0], [1 1], [0 1], [0 2]. Firstly, the user should click the top left button to load this file (See Figure 9.)

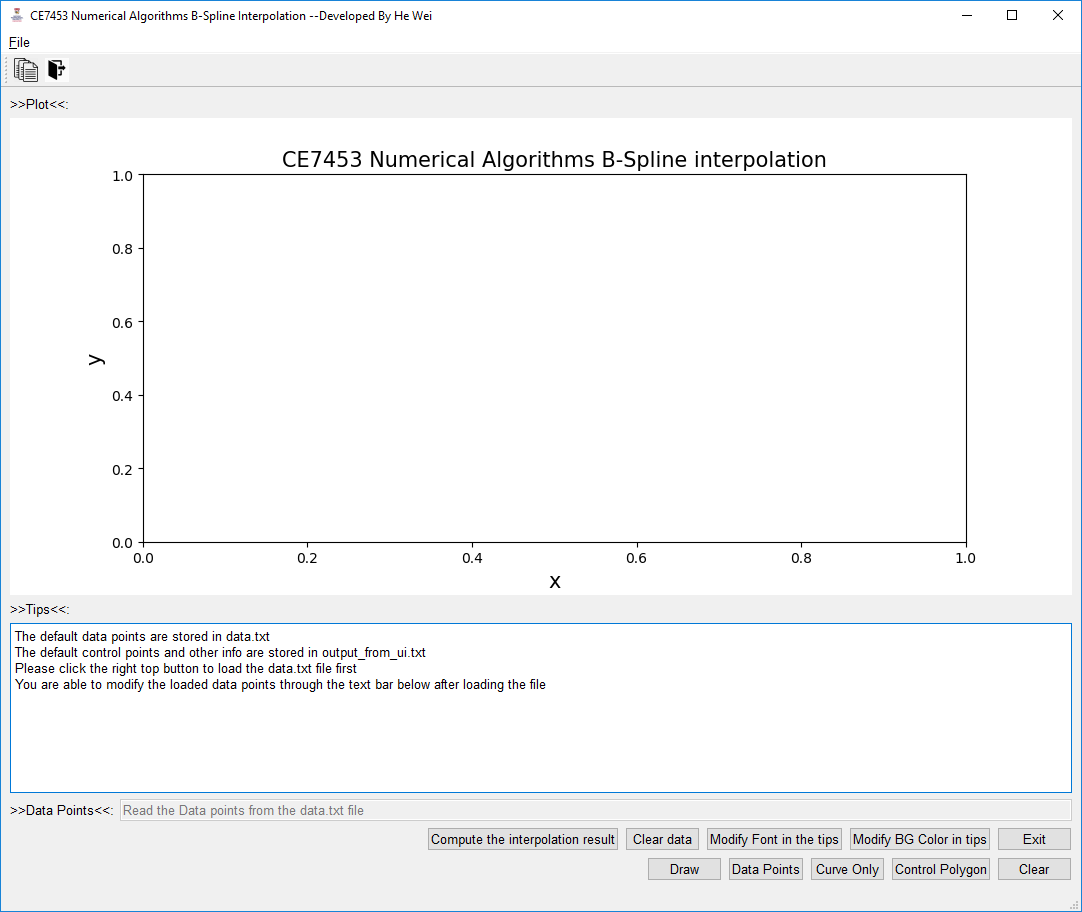


Figure 8. The initial GUI window

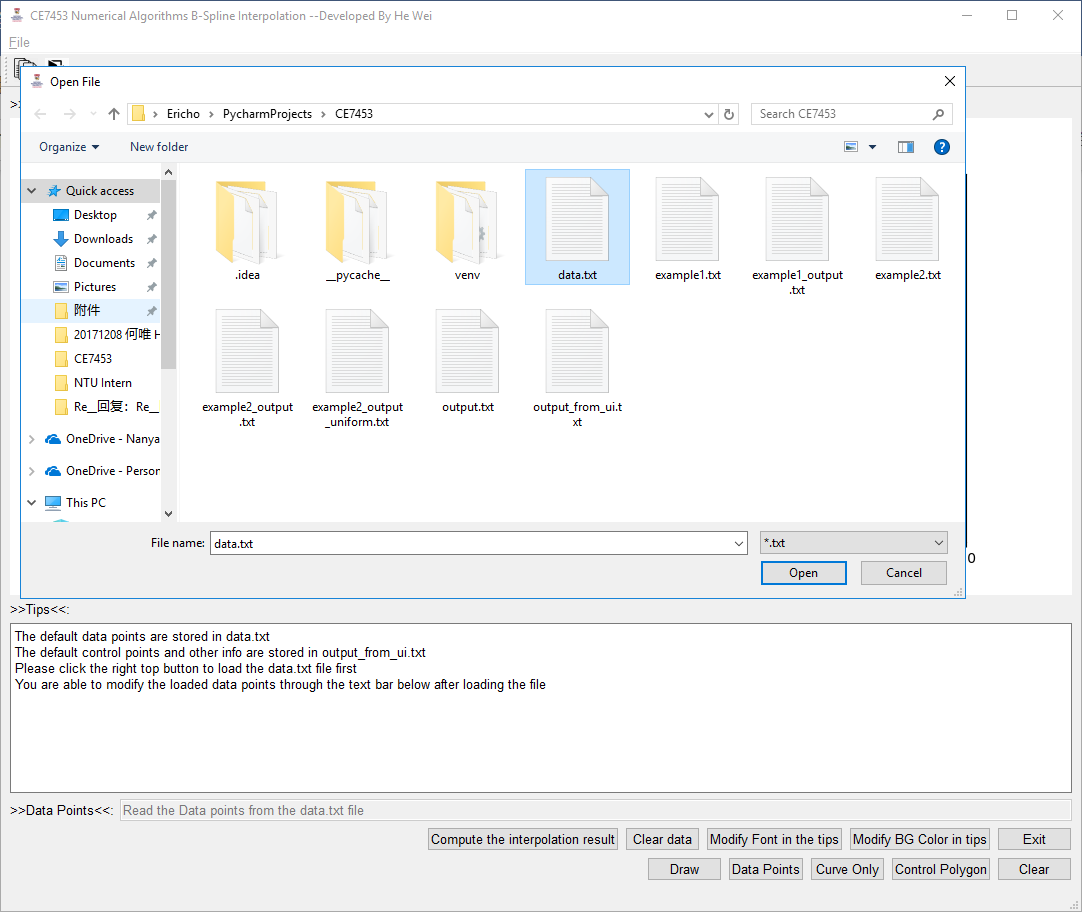


Figure 9. Load the data.txt file

* **Compute the cubic B-spine control points from the data:**

Loading the data file, the program will automatically compute the require information of the cubic B-spline curve (knots, control points and degree).

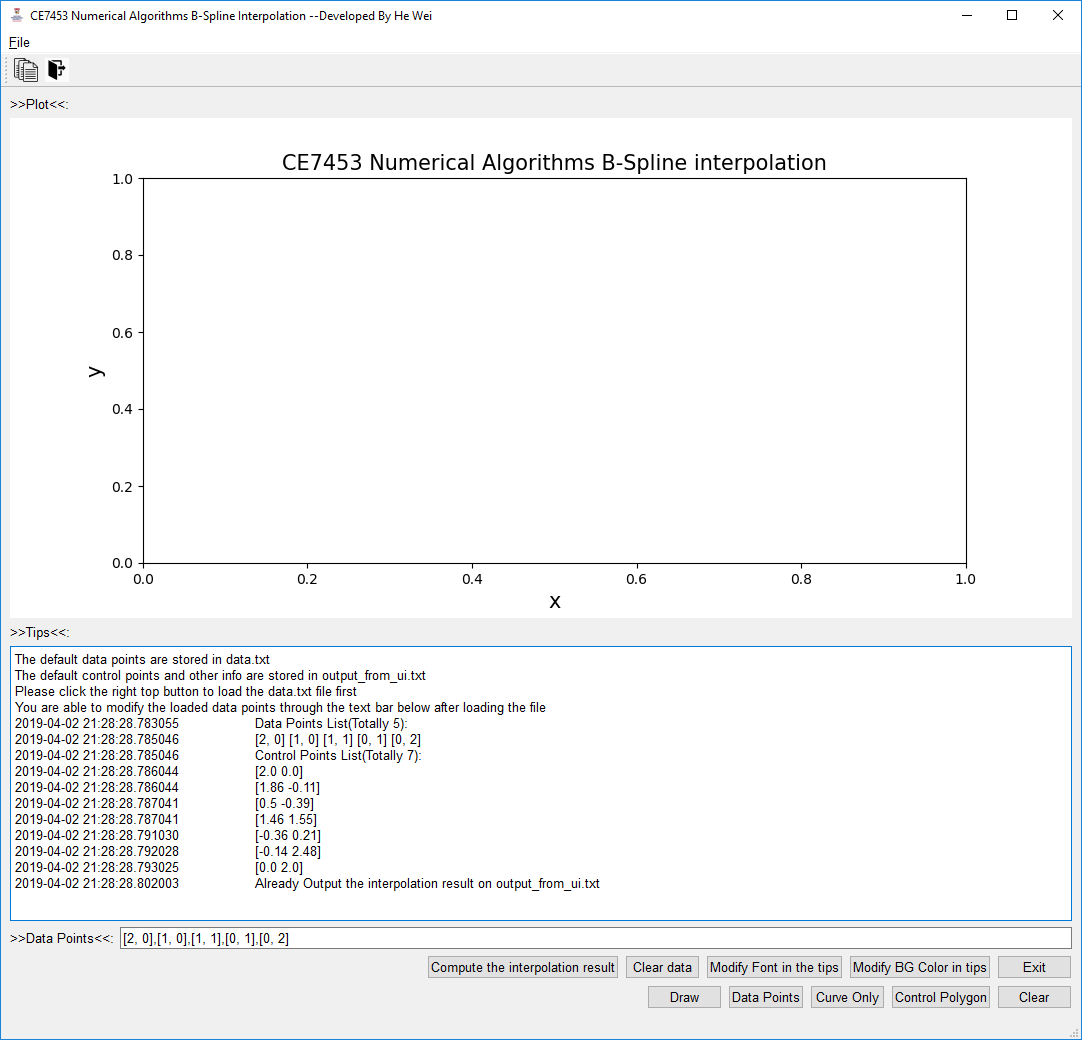


Figure 10. Compute the control points

* **Plot the curve based on the control points**

According to the control points, the program plot the curve using Matplotlib after clicking the draw button. (See Figure 11.) The plot shows the control polygon and the curve in the canvas with different colors.

The program can show the control points scatter diagram, control polygon and the interpolating curve separately. (See Figure 12, Figure 13 and Figure 14)

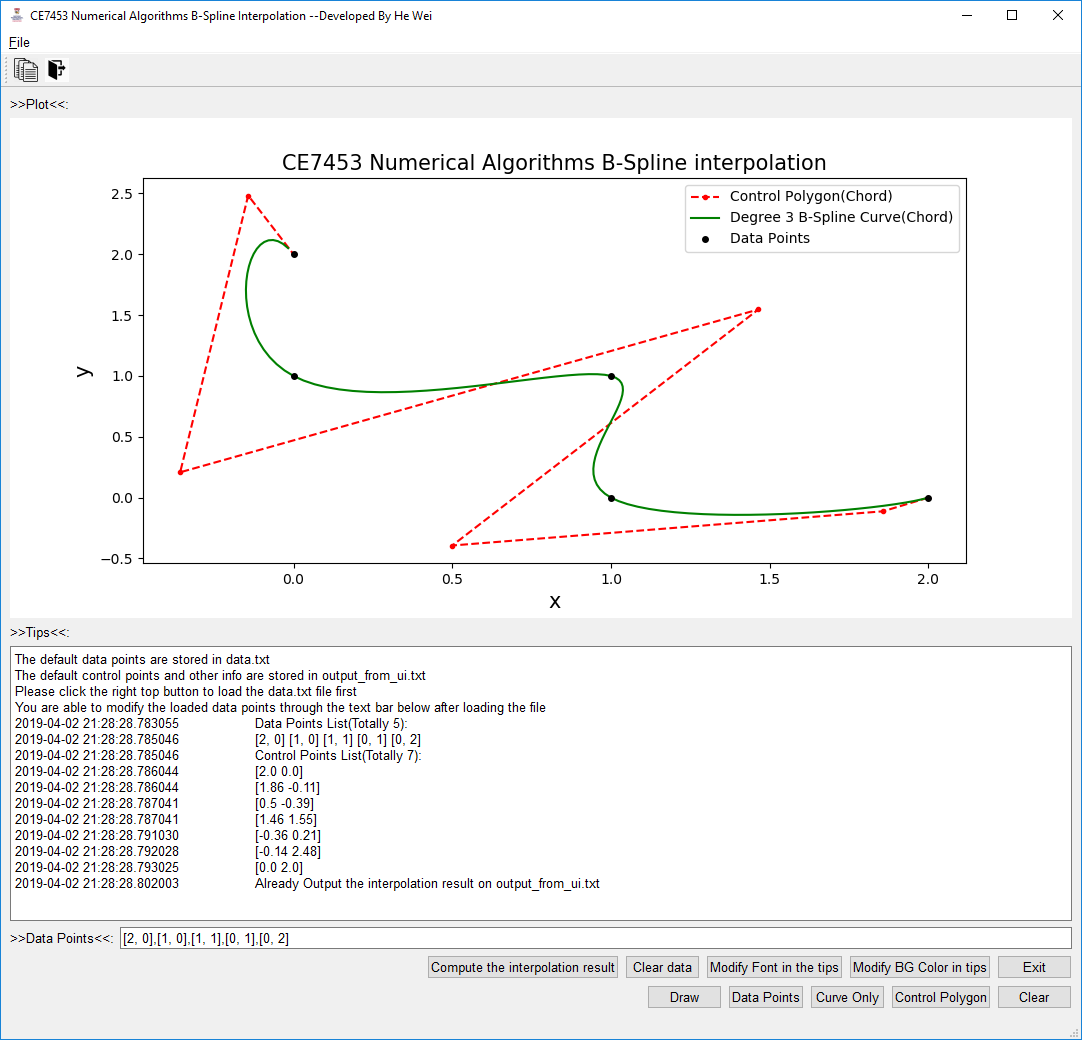


Figure 11. Plot the interpolating curve

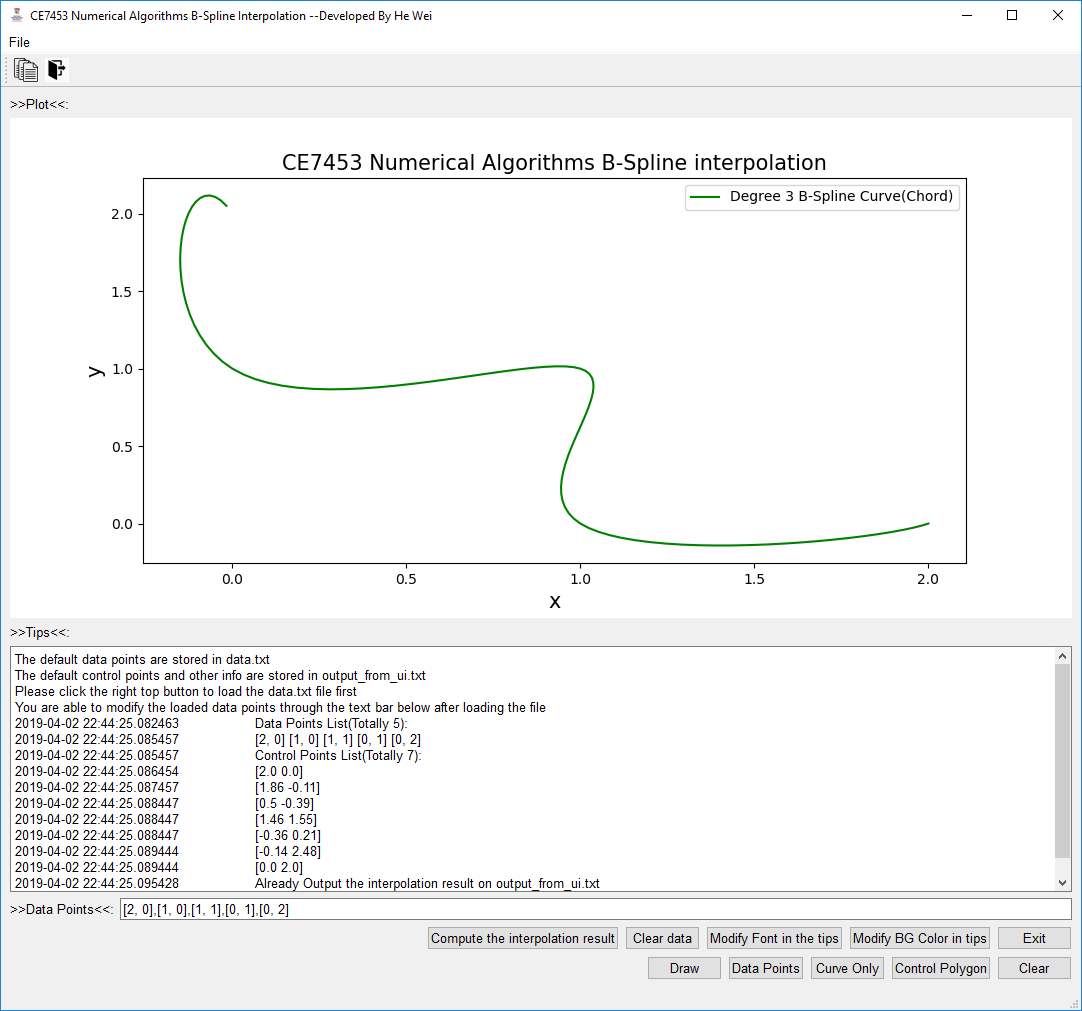


Figure 12. Plot the interpolating curve only

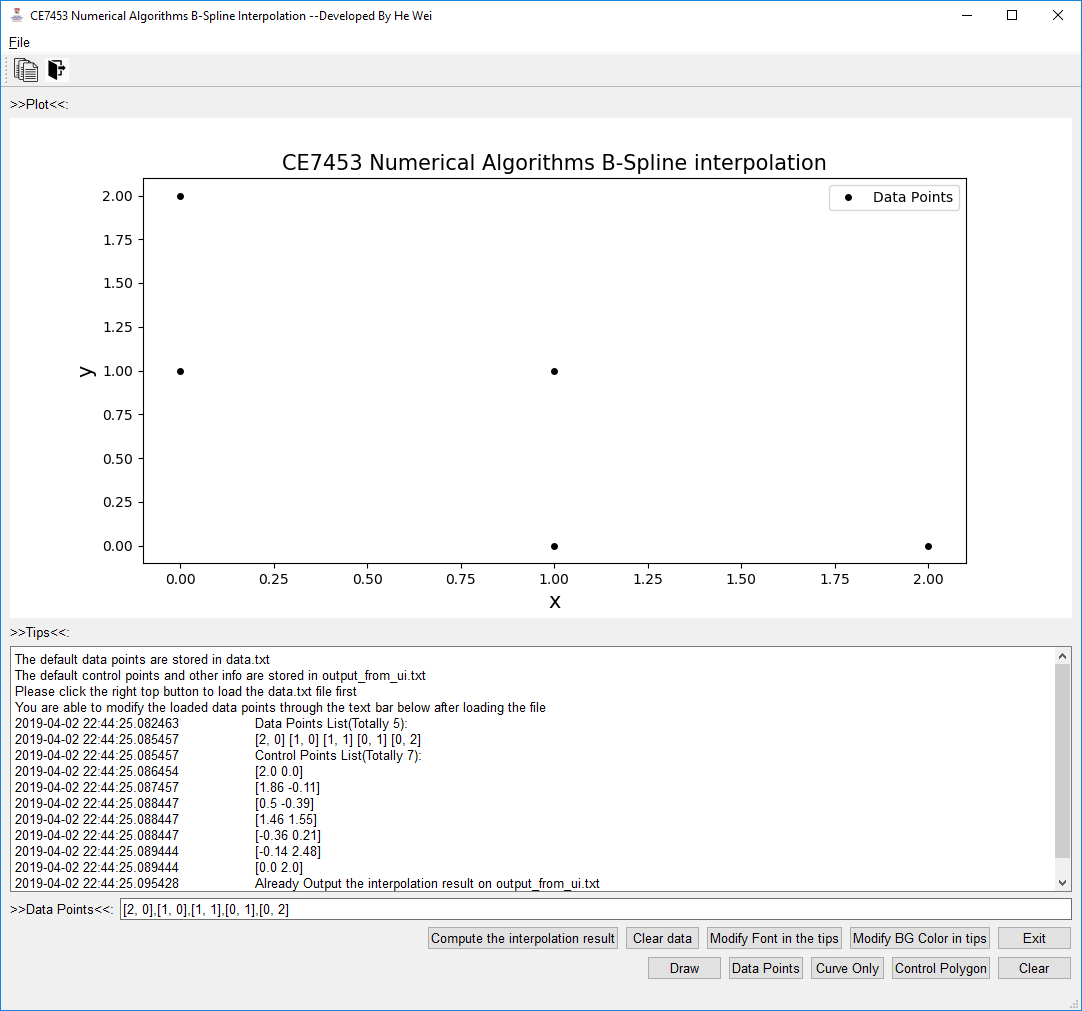


Figure 13. Plot the data points only

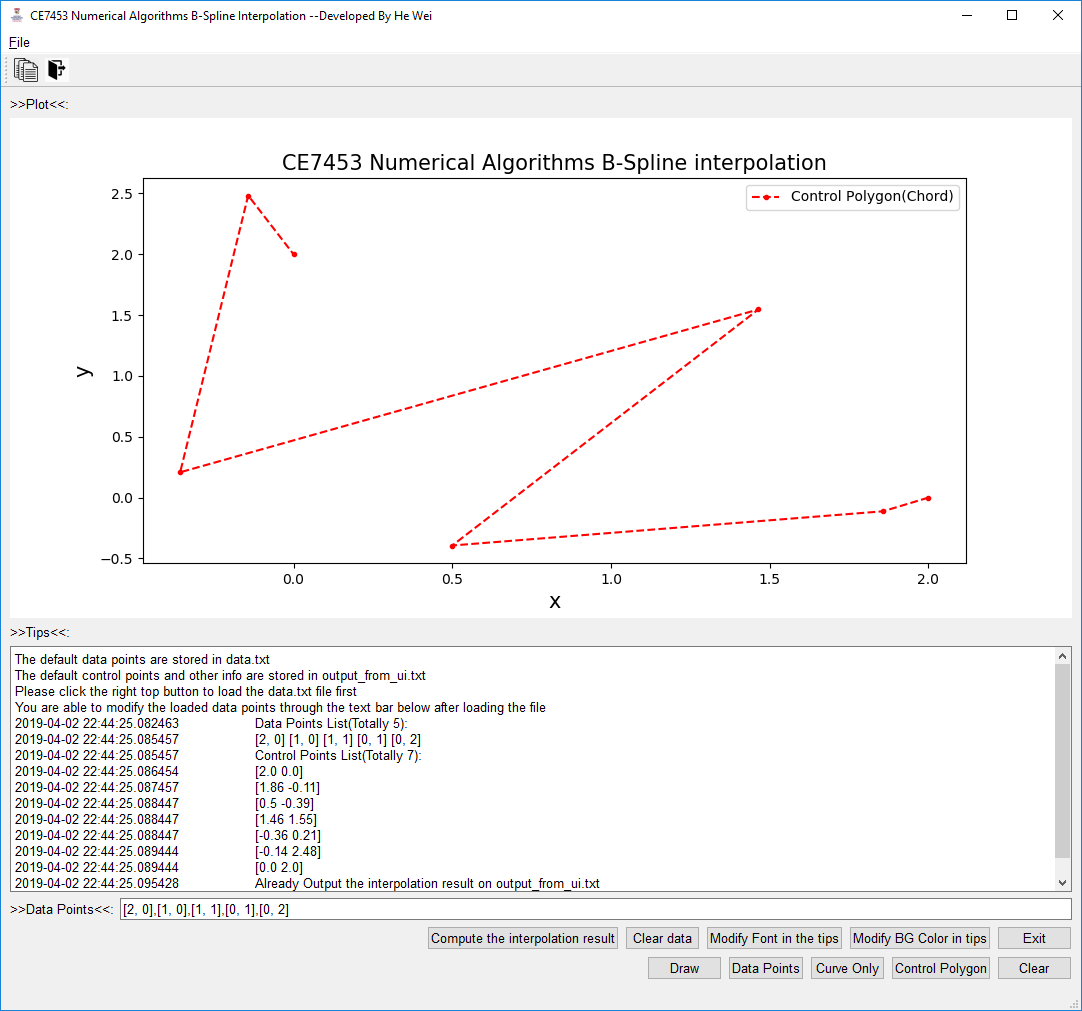


Figure 14. Plot the control polygon only

* **Modify the data points and re-compute the control points**

After plotting the interpolating curve based on the default data points, the user can only modify specific data point coordinate and plot the new interpolating curve. (See Figure 15 and Figure 16)

For example, in Figure 15, the input data points are modified as follows:

[16, 0], [1, 0], [1, 1], [0, 1], [0, 16]

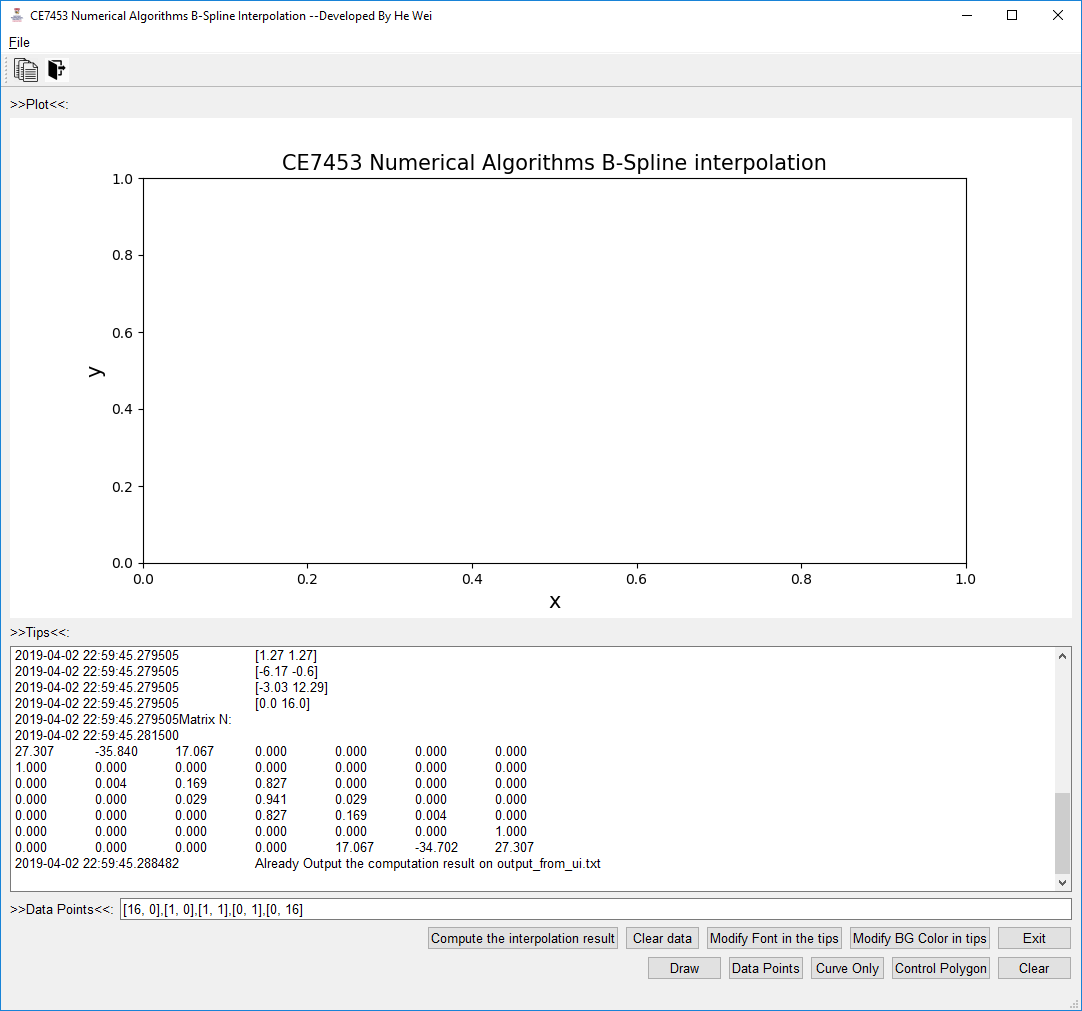


Figure 15. Modify the data points

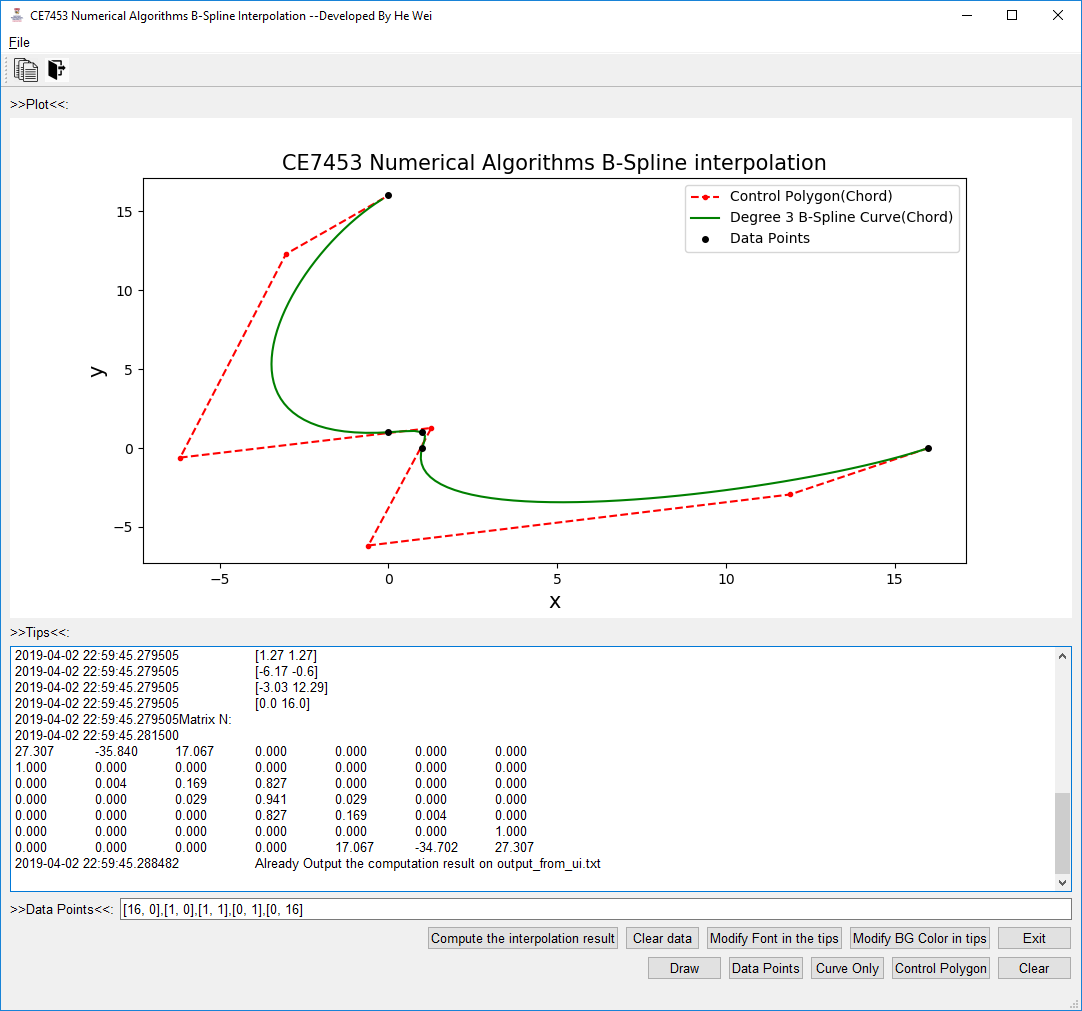


Figure 16. Plotting the new interpolating curve