

# CZ7453 Assignment 1 Report

Wu Ziqing G2105028H ziqing002@e.ntu.edu.sg

April 15, 2022

## Question 1: Implementation of Algorithm

The cubic b-spline interpolation algorithm specified in the lecture is implemented in `main.py`, particularly in `InterpolationAlgo()` class.

The main idea is that for  $n+1$  data points  $P_0 \dots P_n$  to interpolate, we need to compute  $n+3$  control points to generate a cubic b-spline curve. First, we construct  $n+1$  equations based on definition of b-spline curve, to ensure that the curve passes through each point. Then in order to find the exact value of  $n+3$  control points, we add 2 more natural boundary conditions, i.e., the second derivative at two end  $P_0$  and  $P_n$  are 0. The equation system is then solved by a solver provided by the python library.

During the implementation, a few points are reckoned to be worth highlighting:

- For the  $n+1$  equations for each data point, I used the recursive way to find the point coefficient  $N_i^k(t)$  to ensure the code works for any degrees provided. However, for the two endpoint conditions, I have to hard code the second derivative equations specifically for cubic b-splines as there is no explicit recursive formula available for calculating second derivatives for any degrees.
- Due to the existence of multiplicity in knots, when calculating coefficient  $N_i^k(t)$  there might be  $0/0$  cases. I used the trick described in the lecture, where for each knot with repeating value, I progressively add a small  $e = 10^{-5}$  to it. It can effectively solve the  $0/0$  problem.

## Question 2: Linear System Derivation for the Example

In the example, we are required to find the b-spline curve which interpolates points  $(0, 0)$ ,  $(0, 2)$ ,  $(2, 2)$ ,  $(2, 0)$  and  $(4, 0)$ . Based on the result calculated from my program, the linear system for this curve can

be written as in Equation 1.

$$\begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. \\ 96.012 & -144.017 & 48.006 & 0. & 0. & 0. & 0. \\ 0. & 0.25 & 0.583 & 0.167 & 0. & 0. & 0. \\ 0. & 0. & 0.167 & 0.667 & 0.167 & 0. & 0. \\ 0. & 0. & 0. & 0.167 & 0.583 & 0.25 & 0. \\ 0. & 0. & 0. & 0. & 47.997 & -143.986 & 95.988 \\ 0. & 0. & 0. & 0. & 0. & 0. & 1. \end{bmatrix} \begin{bmatrix} (x_0, y_0) \\ (x_1, y_1) \\ (x_2, y_2) \\ (x_3, y_3) \\ (x_4, y_4) \\ (x_5, y_5) \\ (x_6, y_6) \end{bmatrix} = \begin{bmatrix} (0, 0) \\ (0, 0) \\ (0, 2) \\ (2, 2) \\ (2, 0) \\ (0, 0) \\ (4, 0) \end{bmatrix} \quad (1)$$

By solving the linear system, we can get coordinates for all the control points as in Equation 2.

$$\begin{bmatrix} (x_0, y_0) \\ (x_1, y_1) \\ (x_2, y_2) \\ (x_3, y_3) \\ (x_4, y_4) \\ (x_5, y_5) \\ (x_6, y_6) \end{bmatrix} = \begin{bmatrix} (0, 0) \\ (-0.239, 0.786) \\ (-0.716, 2.358) \\ (2.856, 2.569) \\ (1.285, -0.644) \\ (3.095, -0.215) \\ (4.0, 0.0) \end{bmatrix} \quad (2)$$

The output file of the program is shown in Figure 1, showing the knots as well as the control points

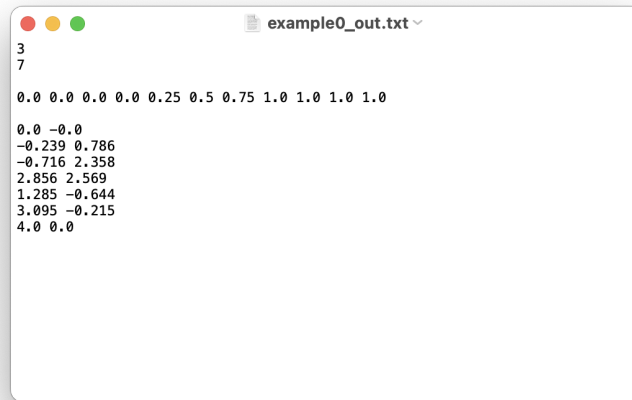


Figure 1: Program Output

The b-spline curve is plotted in Figure 2, where round points are data points, square points are control points and the blue curve is the b-spline curve.

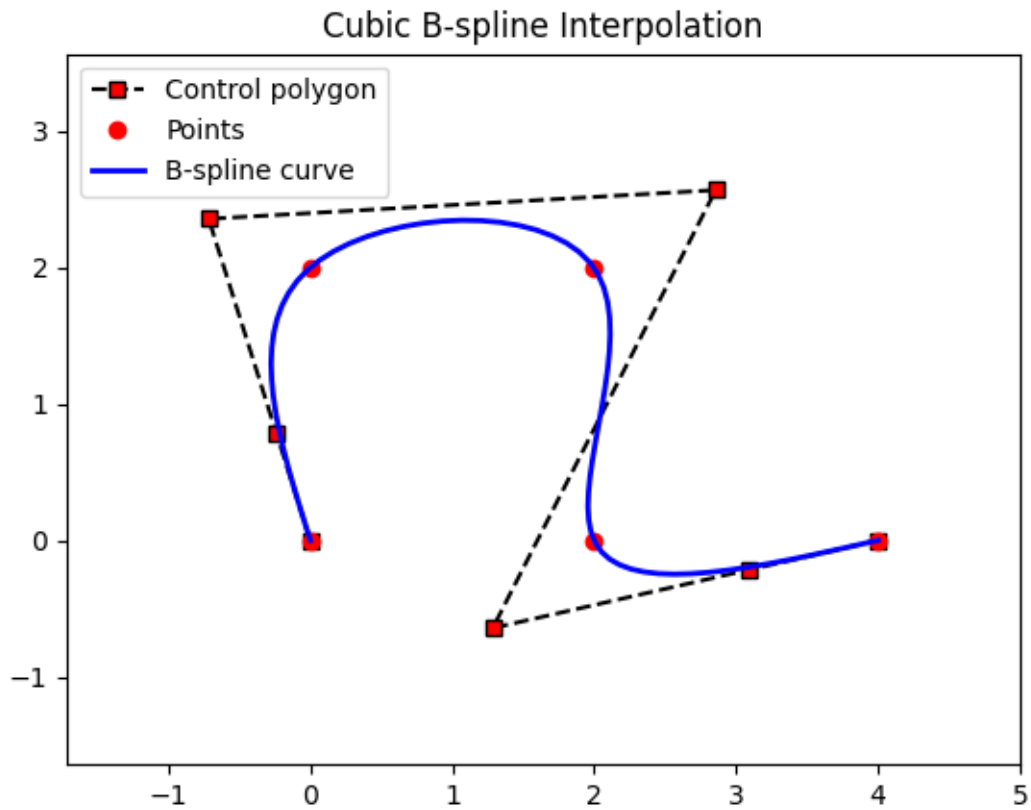


Figure 2: B-Spline Curve for Required Inputs

### Question3: More Interpolation Examples

Two more examples with more than 10 data points are shown in this part.

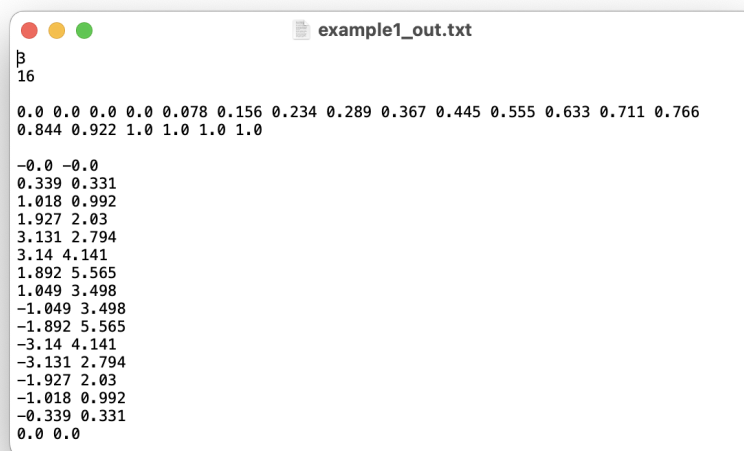
#### Example 1

In this example 14 points are chosen to form a close-loop curve, as shown in Figure 3. The output file is shown in Figure 4, and the plotted graph is displayed in Figure 5.



```
0 0
1 1
2 2
3 3
3 4
2 5
1 4
-1 4
-2 5
-3 4
-3 3
-2 2
-1 1
0 0
```

Figure 3: Example 1 Input



```
p
16
0.0 0.0 0.0 0.0 0.078 0.156 0.234 0.289 0.367 0.445 0.555 0.633 0.711 0.766
0.844 0.922 1.0 1.0 1.0 1.0
-0.0 -0.0
0.339 0.331
1.018 0.992
1.927 2.03
3.131 2.794
3.14 4.141
1.892 5.565
1.049 3.498
-1.049 3.498
-1.892 5.565
-3.14 4.141
-3.131 2.794
-1.927 2.03
-1.018 0.992
-0.339 0.331
0.0 0.0
```

Figure 4: Example 1 Output

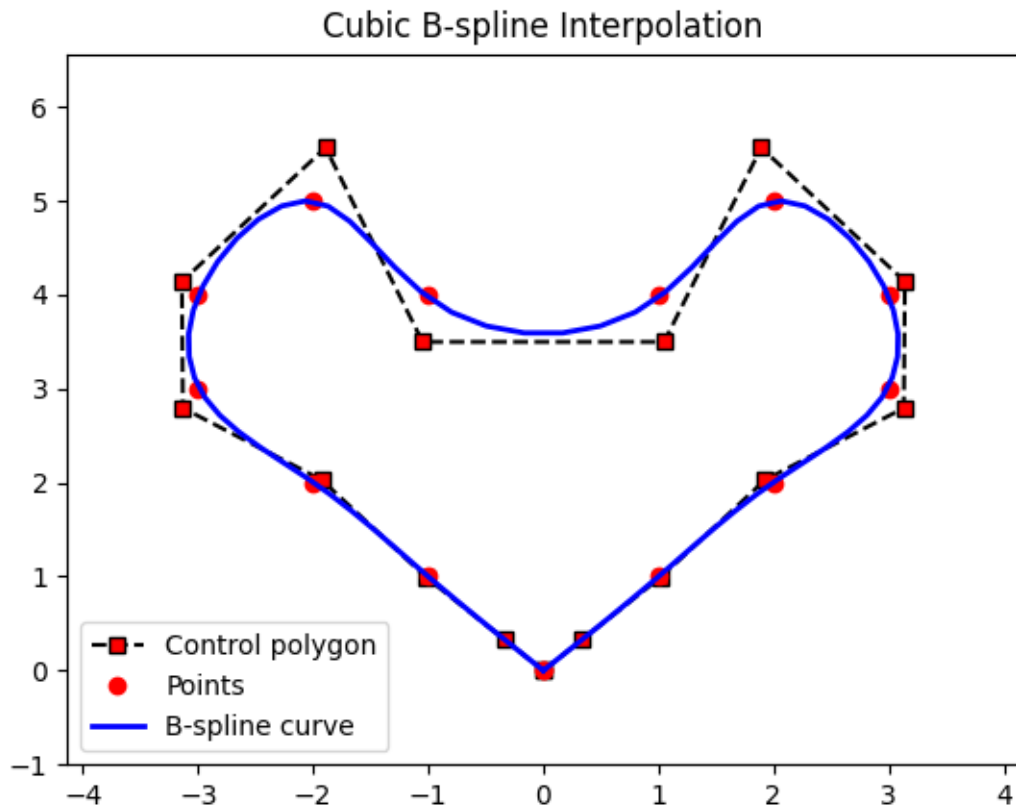
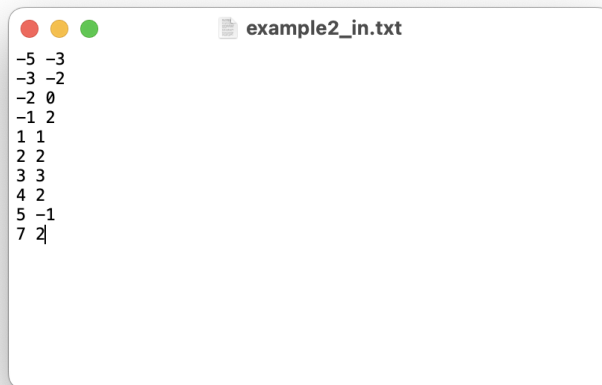


Figure 5: Example 1 Plot

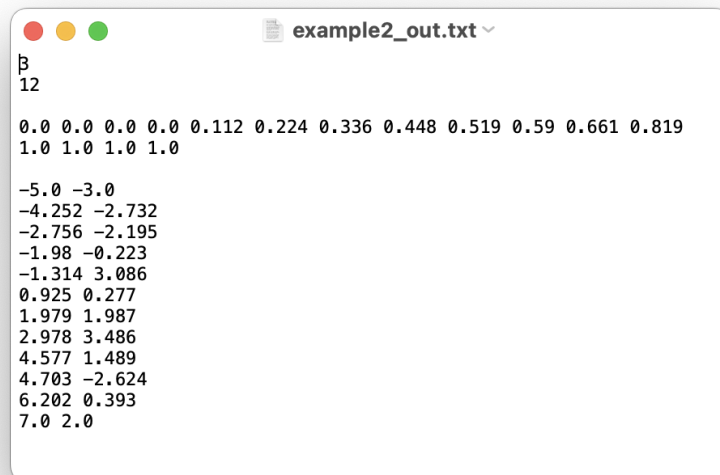
## Example 2

In this example 10 points are chosen to form another b-spline curve, as shown in Figure 6. The output file is shown in Figure 7, and the plotted graph is displayed in Figure 8.



```
example2_in.txt
-5 -3
-3 -2
-2 0
-1 2
1 1
2 2
3 3
4 2
5 -1
7 2
2 1
1 2
```

Figure 6: Example 2 Input



```
example2_out.txt
p
12
0.0 0.0 0.0 0.0 0.112 0.224 0.336 0.448 0.519 0.59 0.661 0.819
1.0 1.0 1.0 1.0
-5.0 -3.0
-4.252 -2.732
-2.756 -2.195
-1.98 -0.223
-1.314 3.086
0.925 0.277
1.979 1.987
2.978 3.486
4.577 1.489
4.703 -2.624
6.202 0.393
7.0 2.0
```

Figure 7: Example 2 Output

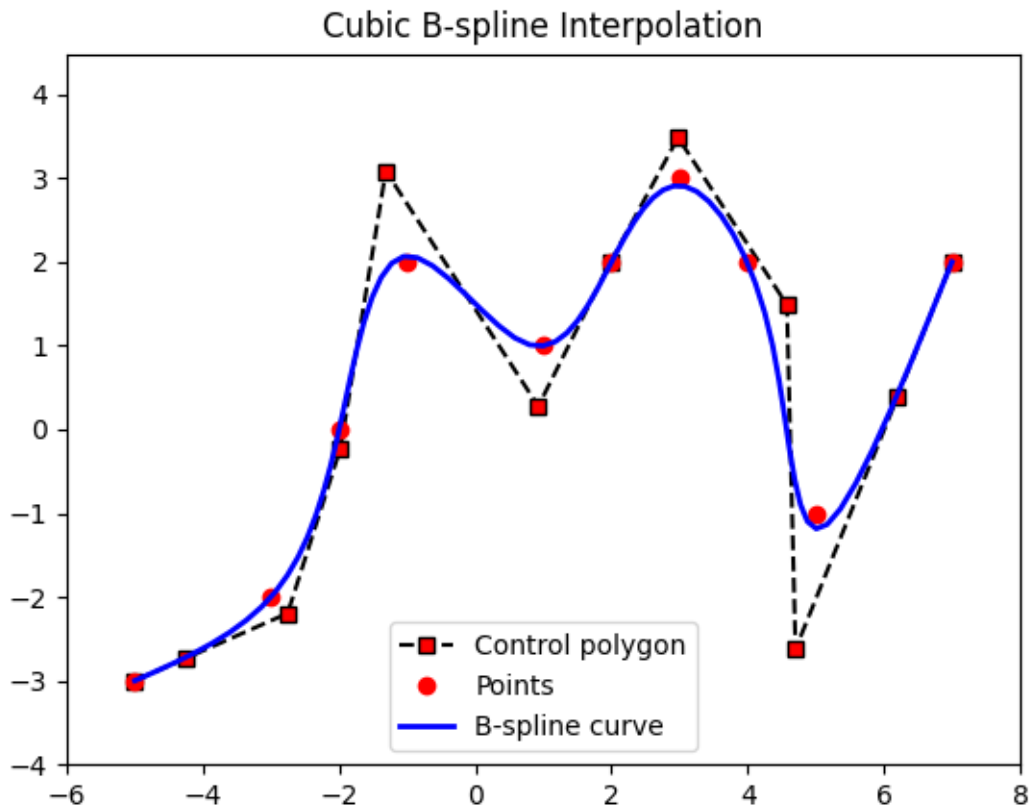


Figure 8: Example 2 Plot

### Question 4: Plotting of B-Spline Curve

The program itself can plot b-spline curves based on the calculated control points, with the help of python libraries. All need to be done is simply to run the code.

### Question 5: Further discussions

1. What if the number of the input data is less than 4?

As long as the data point number  $n > 1$ , the algorithm is able to build a linear system and solve for deterministic  $n + 2$  control points and the curve can still be plotted. Moreover, when  $n = 2$ , the curve will be a straight line. When  $n = 3$ , the curve will be a bezier curve with the third control point as the only control point.

2. What if there are two data points with the same coordinates?

Repeated data points will cause error when solving for the linear system, as empirically suggested by the system.