Numerical Analysis Project

Jinchen Wang

January 2023

1 Introduction

1.1 Definition of the Spline

> **Definition 1** Firstly, we state the boundary condition of those cubic spline. In this project, we only implement those three cubic below.
>
> *natural*: $s''(t_1) = s''(t_N) = 0.$
>
> *complete*: $s'(t_1) = f'(t_1),\ s'(t_N) = f'(t_N).$
>
> *second-derivatives-at-end*: $s''(t_1) = f''(t_1),\ s''(t_N) = f''(t_N).$

1.2 Frame of the class

1.2.1 ppForm

```
ppform(vector<double> _x, vector<double> _y)
void complete_cubic(double m_1, double m_N)
    construct the cubic with the boundart condition:
    s'(t_1) = m_1, s'(t_n) = m_N
void specified_sec_diff(double M_1, double M_N)
    construct the cubic with the boundart condition:
    s''(t_1) = M_1, s''(t_n) = M_n
void natural_cubic(){
    specified_sec_diff(0.0, 0 0.0)
}
double cubic(int _i, double _x)
void Draw_Spline_with_Matlab_Code(int i)
    draw the function
    i is to control the name of the outputfile
void Draw_Error_with_Matlab_Code(int i)
    i is to control the name of the outputfile
```

1.2.2 Bspline

```
 BSpline(vector<double> _x, vector<double> _y, int _k)
 double B(int _i, int _n, double _x)
 double dB(int _i, int _n, double _x)
 double d2B(int _i, int _n, double _x)
void complete(double m_1, double m_N)
    construct the cubic with the boundart condition:
    s'(t_1) = m_1, s'(t_n) = m_N
void specified_sec_diff(double M_1, double M_N)
    construct the cubic with the boundart condition:
    s''(t_1) = M_1, s''(t_n) = M_n
void natural_cubic(){
    specified_sec_diff(0.0, 0 0.0)
}
double compute(double _x)
    compute the function at x = _x
double error(double _x)
    compute the error at x = _x
void Draw_Spline_with_Matlab_Code(int i)
    draw the function
    i is to control the name of the outputfile
void Draw_Error_with_Matlab_Code(int i)
    i is to control the name of the outputfile
```

2 How to test

Test in Windows.

```
mingw32-make
```

It will produce several MATLAB programs. The MATLAB program needs to be run manually.
Remark:
There is something run with the inherit of the Function Class. So I manually define 3 Function Class.

3 ProblemA

HOW TO RUN

```
./testA.exe
 N = 6,11,21,41,81
```

3.1 Plot Function

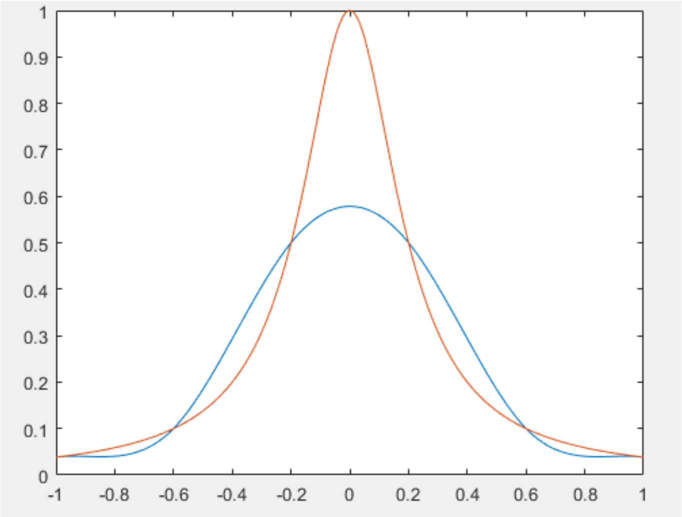

Figure 1: 6 knots

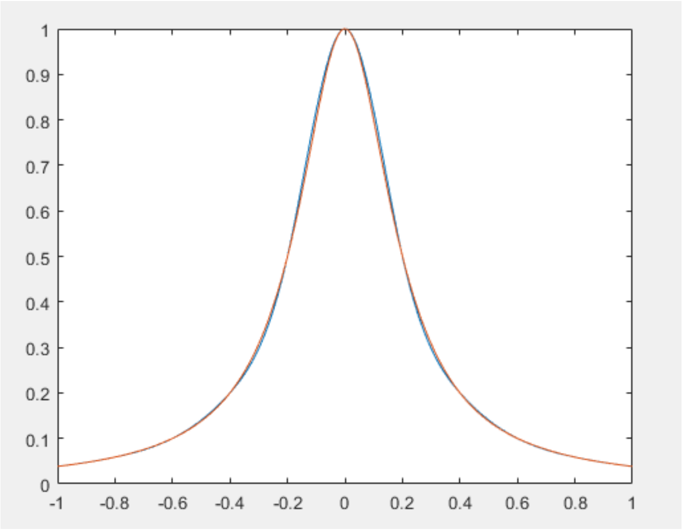

Figure 2: 11 knots

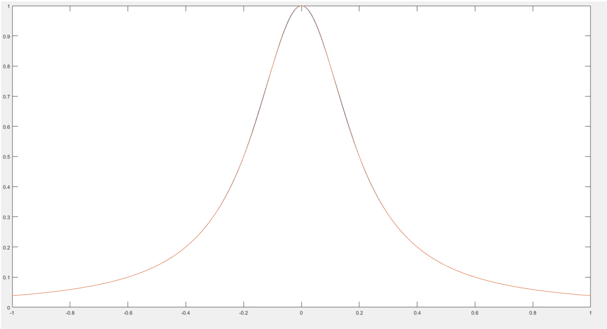

Figure 3: 21 knots

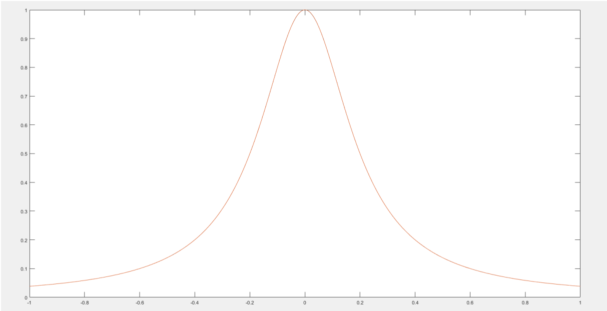

Figure 4: 41 knots

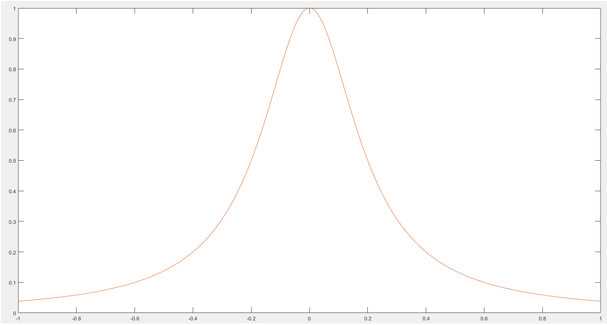

Figure 5: 81 knots

3.2 Error Analysis

N	6	11	21	41	81
Error	0.4217	0.0205289	0.00316894	2.7536e-04	1.6090e-05

4 ProblemB

HOW TO RUN

```
./testBCD.exe
N=10
a=−5,−4.5
b=5.4.5
```

4.1 Plot Function

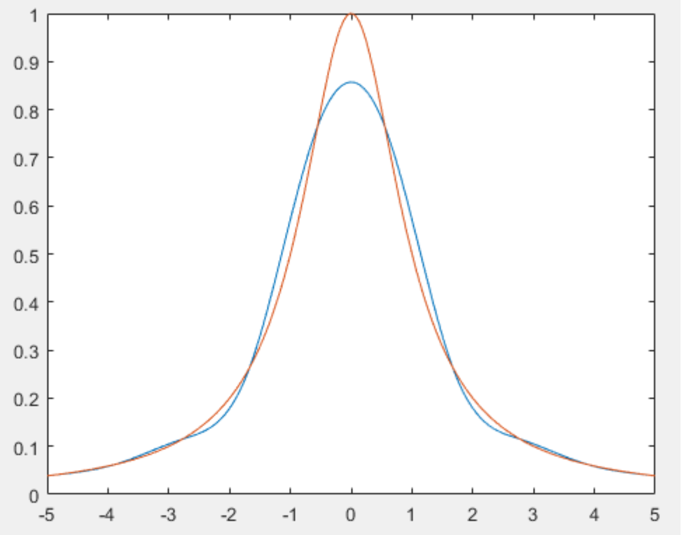

Figure 6: $t_i = -6 + i$

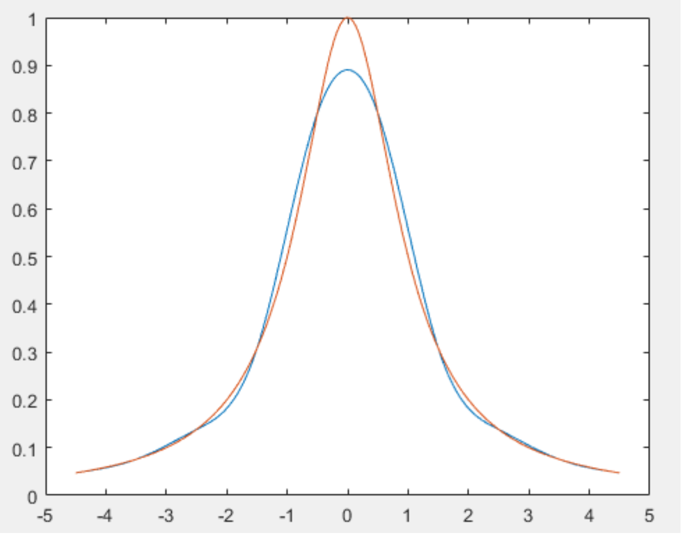

Figure 7: $t_i = -\frac{11}{2} + i$

4.2 Plot error

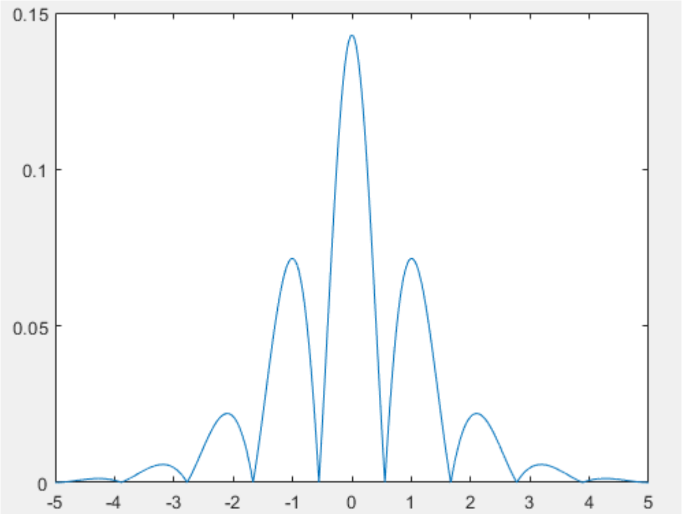

Figure 8: $t_i = -6 + i$

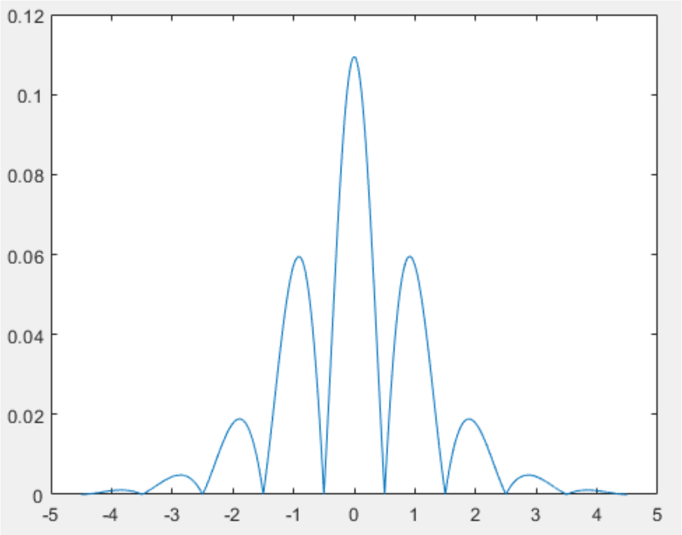

Figure 9: $t_i = -\frac{11}{2} + i$

4.3 Error Analysis

Knots	-3.5	-3	-0.5	0	0.5	3	3.5
Error	1,00389263	0.00469698	0.0181832	0.142871	0.0181832	0.00469698	0.00389263
Error	0	0.00443972	0	0.109398	0	0.00443972	0