Numerical Analysis HW10 : Spline Interpolations

Ming-Chang Chiu 100060007

May 12, 2015

1 Objective

In this assignment, we are given different numbers of support points in each .dat file, with $475 \leq x \leq 775$. We are required to use Spline interpolation to interpolate the Y values for each $475 \leq x \leq 775$ and then observe the results as well as the difference between given values and interpolated values. I implemented

void splineM(int N,VEC &X,VEC &Y,VEC &M)

double spline(double x,int N,VEC &X,VEC &Y,VEC &M)

as my interpolating functions, splineM() to generate spline momentum, and spline() to calculate the interpolated value.

2 Implementation of cubic Spline

Cubic Spline is defined as follow:

Given a partition Δ of $[a, b]$ and a set of support points $\{(x_i, y_i), i = 0, 1, \ldots, n\}$, a cubic spline S_Δ on Δ is a real function $S_\Delta : [a, b] \to \mathbb{R}$ with the following properties:

1. $S_\Delta \in C^2[a, b]$, that is, S_Δ is twice continuously differentiable on $[a, b]$.
2. S_Δ coincides on every subinterval $[x_{i-1}, x_i]$, $i = 1, 2, \ldots, n$, with a polynomial of degree at most three.

and we further assume

$\{(x_i, y_i), i = 0, 1, \ldots, n\}$ are the support points,
$\Delta = \{x_i, i = 0, 1, \ldots, n\}$ is the partition,
$I_i = [x_{i-1}, x_i]$, $i = 1, \ldots, n$ are the subintervals,
and $h_i = x_i - x_{i-1}$ is the length of the subinterval

Since S_Δ is twice continuously differentiable, we can have

$$S''_\Delta(x) = M_{i-1}\frac{x_i - x}{h_i} + M_i\frac{x - x_i}{h_i}$$

Integrating the above equation, we have

$$S'_\Delta(x) = -M_{i-1}\frac{(x_i - x)^2}{2h_i} + M_i\frac{(x - x_{i-1})^2}{2h_i} + A_i. \tag{5.2.5}$$

$$S_\Delta(x) = M_{i-1}\frac{(x_i - x)^3}{6h_i} + M_i\frac{(x - x_{i-1})^3}{6h_i} + A_i(x - x_{i-1}) + B_i. \tag{5.2.6}$$

In splineM(), I am computing the M_i, M_{i-1} needed for (5.2.6) and then provide them to spline(), which exactly do what (5.2.6) is doing. In my code, I implement zero boundary moments as constraint in splineM(), which is $M_0 = 0$ and $M_n = 0$. Then we can get the following system of equations to solve for all moments.

$$\begin{bmatrix} 2 & \lambda_0 & 0 & 0 & \cdots & & 0 \\ \mu_1 & 2 & \lambda_1 & 0 & \cdots & & 0 \\ 0 & \mu_2 & 2 & \lambda_2 & \cdots & & 0 \\ & \cdots & & & \ddots & & \vdots \\ & \cdots & & & \cdots & 2 & \lambda_{n-1} \\ 0 & 0 & \cdots & & \cdots & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix} \tag{5.2.24}$$

where μ_i, λ_i and d_i, are given by equations (5.2.18), (5.2.19) and (5.2.20) for $i = 1, 2, \ldots, n-1$ and

$$\begin{aligned} \lambda_0 &= 0, \\ d_0 &= 0, \\ \mu_n &= 0, \\ d_n &= 0. \end{aligned}$$

where

$$\mu_i = \frac{h_i}{h_i + h_{i+1}}, \tag{5.2.18}$$

$$\lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}}, \tag{5.2.19}$$

$$d_i = \frac{6}{h_i + h_{i+1}}\left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right). \tag{5.2.20}$$

3 Workflow

Usage: ./hw10.out * < f*.dat, where * could be 3, 5, 7, 13, 21. For example, ./hw10.out 3 < f3.dat

Solve: Spline method is applied.

Desired output: The program will print the maximal error in [475,775] and build a file named *data.txt*, which stores value of x_i, interpolated value, error at x_i, and original value at x_i in f301.dat sequentially each row. One can load the .txt file for further analysis.

4 Results

File	Spline's max error	Lagrange's max error	Lagrange's max error in [500,700]
f3.dat	354.947325	372.8669	372.8669
f5.dat	190.820112	248.3406	233.3644
f7.dat	73.743612	379.1073	147.5782
f13.dat	29.044764	1283.4489	39.6189
f21.dat	19.641670	16728.5648	17.804

5 Plot Analysis

The error is defined as $|Spline(x_i) - f301(x_i)|$

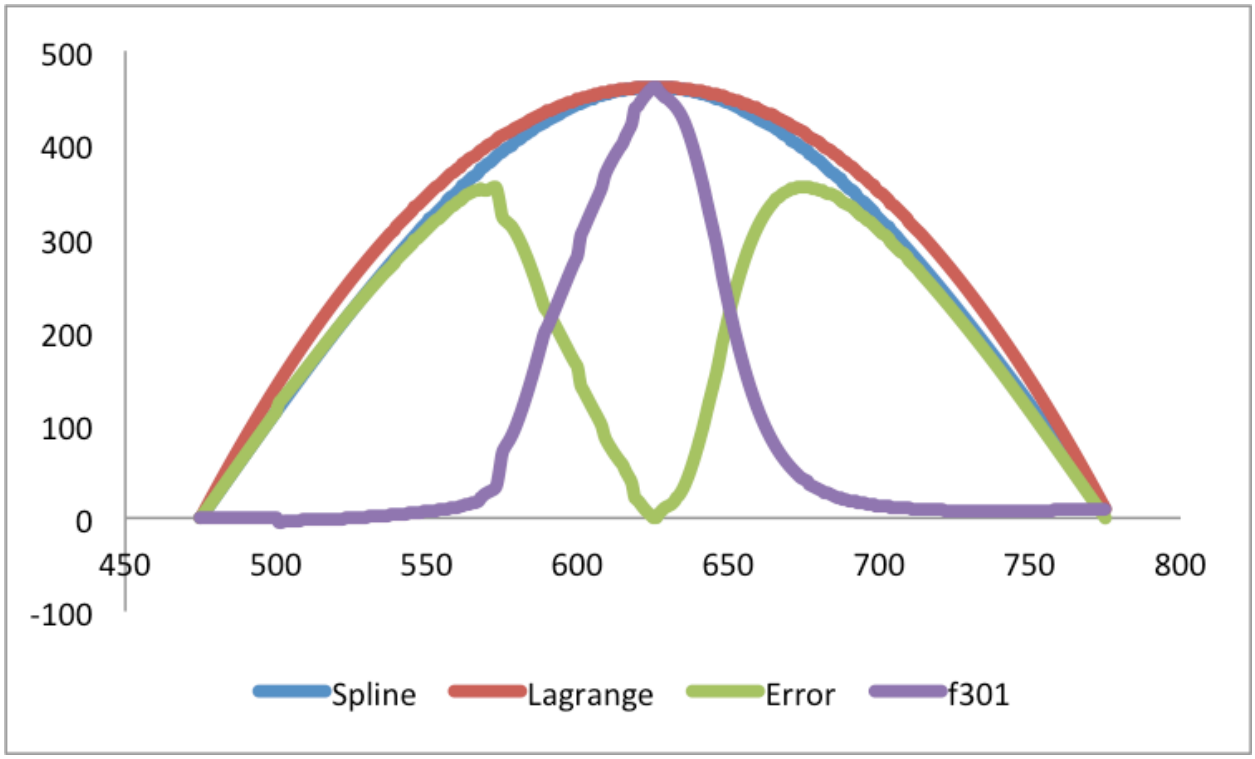

Figure 1: Interpolation with 3 support points

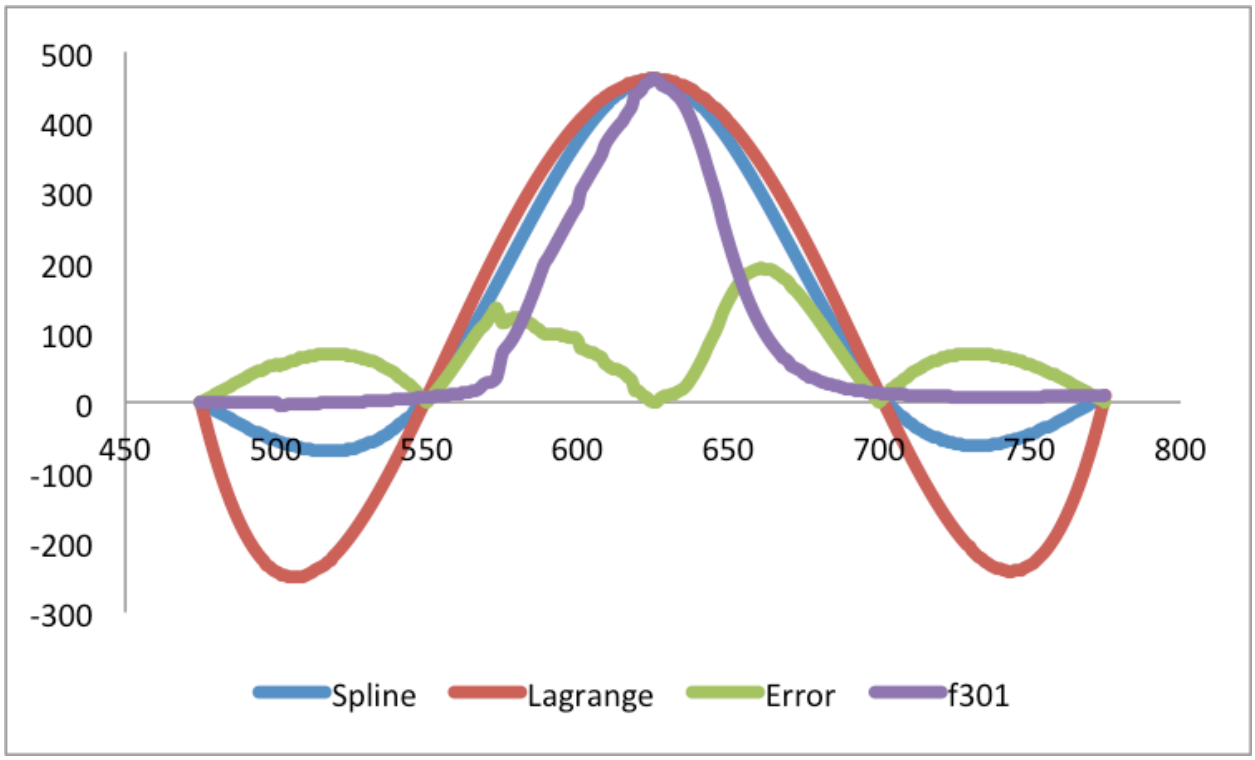

Figure 2: Interpolation with 5 support points

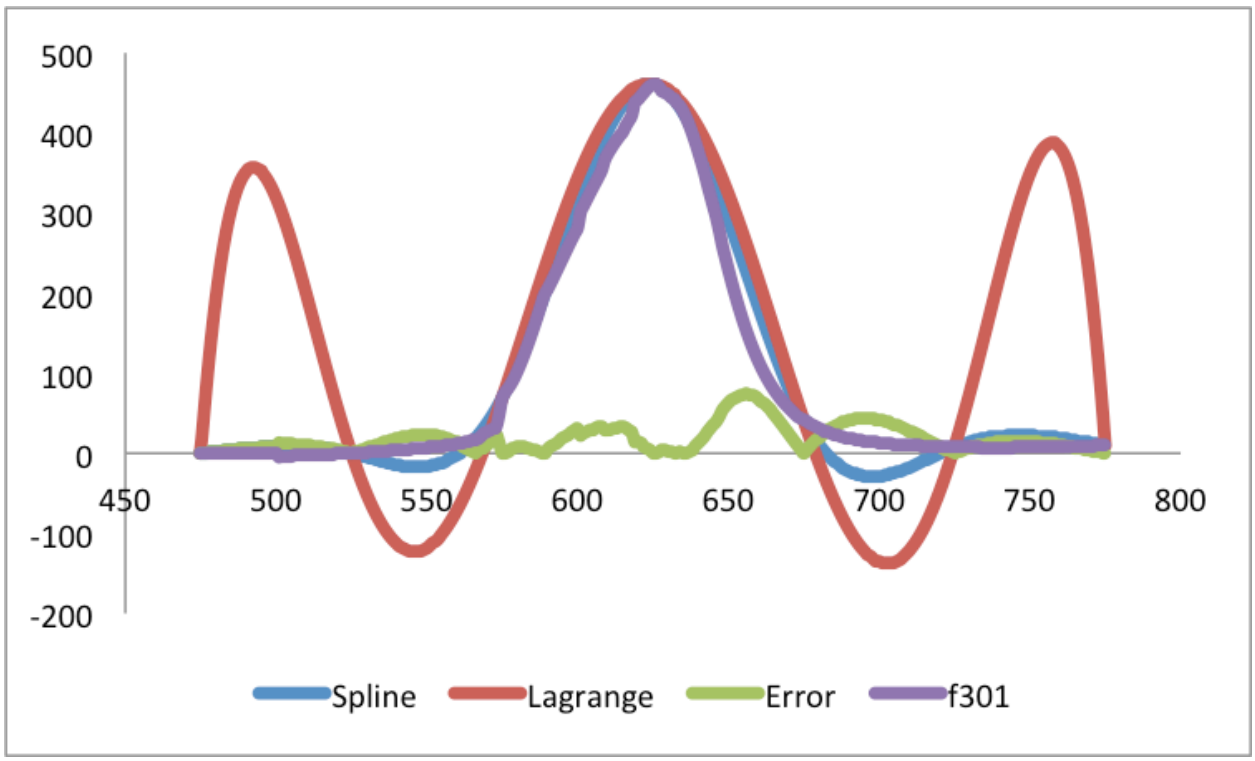

Figure 3: Interpolation with 7 support points

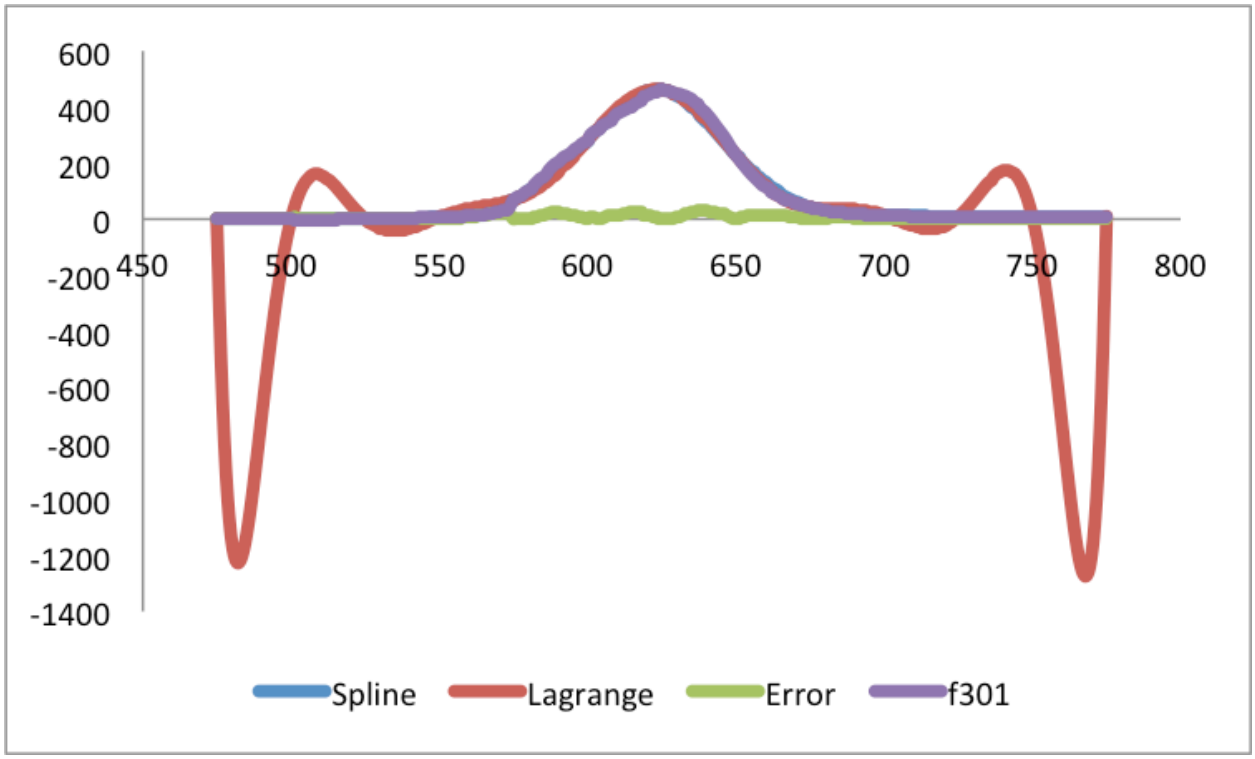

Figure 4: Interpolation with 13 support points

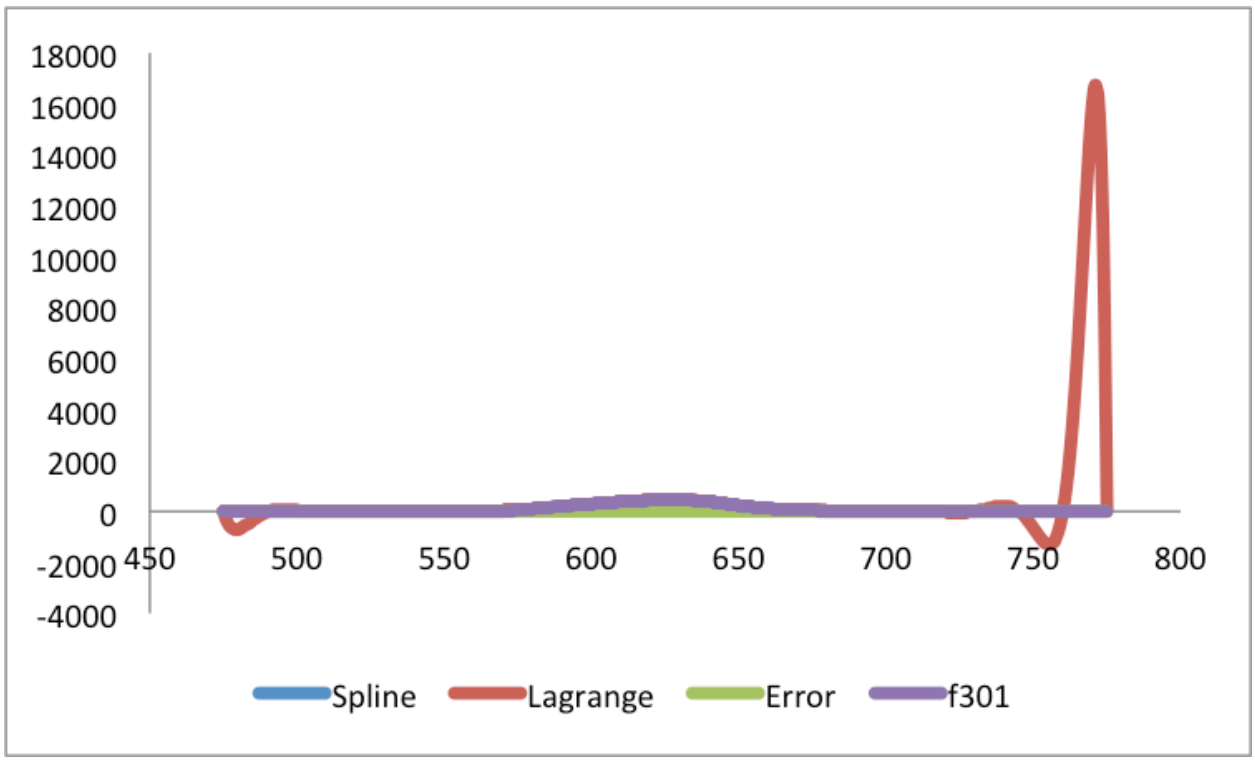

Figure 5: Interpolation with 21 support points

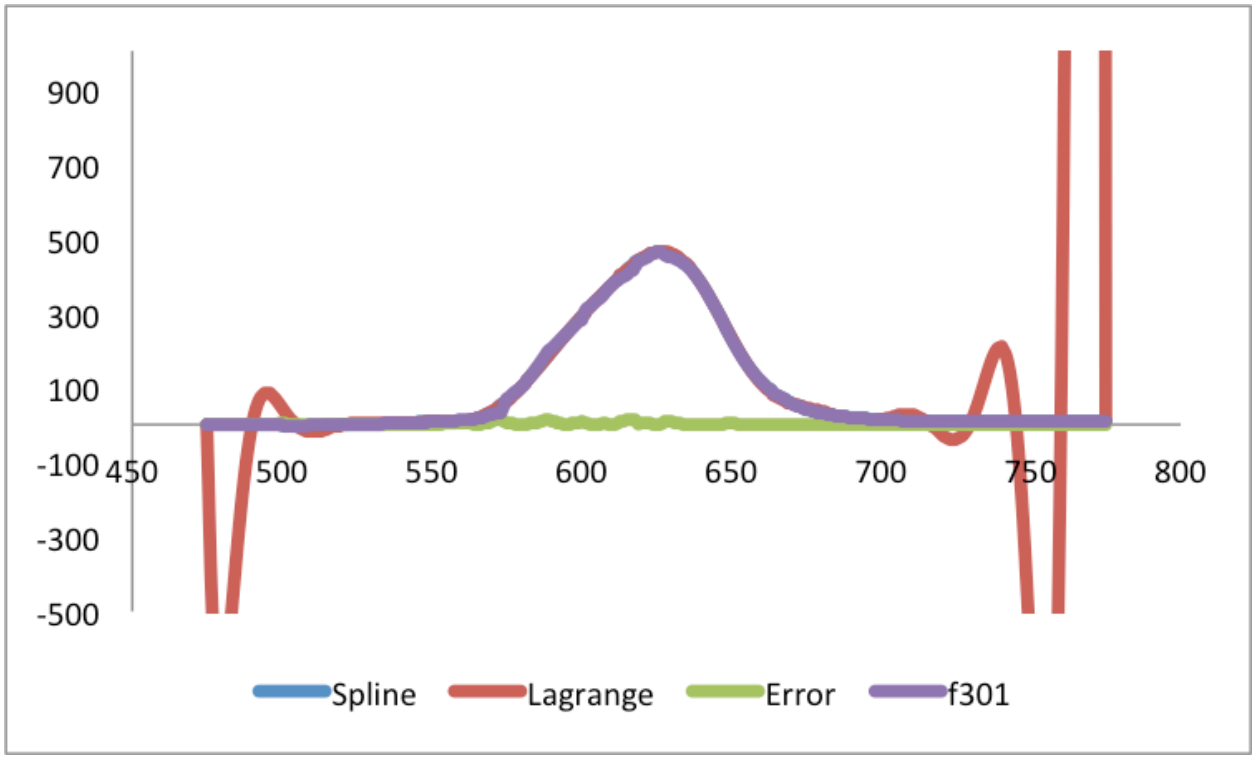

Figure 6: Interpolation with 21 support points: Magnified

6 Observations

In general, I would anticipate that with more support points, the more approximate the interpolated waveform will be. The result tells me my expectation is exactly true.

Compare Lagrange with Spline, we can easily observe that waveforms of Spline are a lot smoother and have massively less error at larger or small x. This is probably because I am using cubic spline interpolation, the order of interpolation formula is at most 3, first and second derivatives of formulae are continuous in each interval, there are much less concave and convex points in the waveforms of Spline than that of Lagrange. With smoother waveform, the error cannot be so large as Lagrange, so Spline's maximal error is then smaller.

In sum, from the table in Section 4, for all range of [475,775] the error is decreasing as more and more support points given, which means Spline will be more and more approximate to the original waveform if large amounts of support points are provided. The reason may reside in that since 1st and 2nd derivatives of cubic Spline functions are continuous, the interpolated waveform will be smooth, with more equal spaced subintervals(i.e. more support points) the interpolated values will be more approximate to f301.