MACM 316 Lecture 24 - Chapter 3

Alexander Ng

Wednesday, March 12, 2025

Even if we enforce the constraint that f and f' agree at the nodes, we still have the problem that P_n will be expensive to compute for large n and that for high n, the interpolating polynomial can still oscillate wildly.

1 Splines (15.1)

Previously, we used a single polynomial to compute an approximation to a function over some finite interval. We increased the degree of the polynomial if we wanted more accuracy. However, increasing the degree of the polynomial causes oscillatory behaviour, and high degree polynomials have the property that a fluctuation over a small portion of the interval can induce large fluctuations over the entire interval.

An alternative approach is to divide the interval into subintervals and use a different, lower degree polynomial in each subinterval. These polynomials are patched together to give a piecewise polynomial approximation.

1.1 Possible Choices

1.1.1 Piecewise Linear Interpolation

We join a set of data points by a series of straight lines.

The biggest disadvantage of this approach is that the approximation is typically not smooth. **i.e.** it is not differentiable at these points, which may not be satisfactory.

1.1.2 Piecewise Polynomails of Hermite Type

*Hermite is pronounced "her-meat"

If we know the value of a function **and** it's derivative at each of the data points, then we could use a Hermite cubic polynomial on each interval $[x_i, x_{i+1}]$ to approximate the function. Often, however, the derivative of the function is not known at the data points.

1.1.3 Piecewise Quadratic Polynomials

Alternatively, we could join a set of polynomials with quadratic polynomials. This gives us 3 arbitrary constants. There will be 2 conditions to fit the curve through the endpoints of each interval, but there isn't enough flexibility to set conditions on the derivative at both endpoints x and x_n.

We have enough flexibility to set conditions for the first $n-1$ data points, but the data point x_n requires four conditions, when we can only provide three degrees of freedom.

- P_0: 2 endpoint conditions + 1 slope condition
- P_1: 2 endpoint conditions + 1 slope conditions
- P_n: 2 endpoint conditions + 2 slope conditions

1.1.4 Cubic Spline Interpolation

If we use cubic polynomials between each successive pair of nodes, then there are four constants and it is possible to ensure that the interpolant

- agrees with the function at all the nodes
- is continuously differentiable
- has continuous second derivatives

Given a function f defined on $[a, b]$ and a set of nodes $a = x_0 < x_1 < \cdots < x_n = b$, a cubic spline interpolant S for f is a function that satisfies the following:

- $S(x) = S_j(x)$ on $[x_j, x_{j+1}]$

- $S_j(x_{j+1}) = f(x_{j+1}) = S_{j+1}(x_j)$ splines must pass through the data points
- $S'_j(x_{j+1}) = S'_{j+1}(x_{j+1})$ first derivative is continuous
- $S''_j(x_{j+1}) = S''_{j+1}(x_{j+1})$ second derivative is continuous

We also need a boundary condition. Typically one of the following hold in the problems we will consider:

- $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ "Clamped Boundary Conditions"
- $S''(x_0) = 0 = S''(x_n)$ "Free or Natural Boundary Conditions"
  *Natural Boundary Conditions tell us that S is not curved at the endpoints.

Of course, for the clamped case, we need derivative information, either from the physics or from some assumption. If there are $(n+1)$ points, the number of intervals and the number of $S_i(x)$'s are n :

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3.$$

Letting $h_i = x_{i+1} - x_i$ we can simplify by substituting x_{i+1} and x_i into $S_i(x)$, $S'_i(x)$ and $S''_i(x)$. Eliminating the b_i and d_i gives a linear system of equations:

$$\begin{aligned} &h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_0 c_{i+1} \\ &\quad = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1}) \\ &\quad 1 \le i \le n-1 \end{aligned}$$

where the a_i are known

$$a_i = f(x_i) \qquad 0 \le i \le n-1.$$

and we define

$$\begin{aligned} a_n &\equiv f(x_n) \\ b_n &\equiv f'(x_n) \\ c_n &\equiv \frac{f''(x_n)}{2} \end{aligned}$$

The b_i and d_i are easily found:

$$b_i = \frac{1}{h_i}(a_{i+1} - a_i) - \frac{h_i}{3}(2c_i + c_{i+1}) \quad 0 \le i \le n-1.$$

$$d_i = \frac{c_{i+1} - c_i}{3h_i}.$$

We still need to impose the boundary conditions:

Natural Boundary Conditions Consider the Natural BC's: $S''(x_0) = S''(x_n) = 0$.

$$\therefore C_n = 0 \quad \text{(by definition)}.$$

$$S_0''(x) = 2C_0 + 6d_0(x - x_0) \therefore S''(x_0) = 2c_0 + 6d_0(x_0 - x_0) = 0 = c_0.$$

We can derive a matrix equation for the $[c_i]$:

$$Ax = b.$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & \dots & \dots & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & \ddots & & \vdots \\ 0 & h_1 & 2(h_1 + h_2) & h_2 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ 0 & \dots & \dots & 0 & 0 & 1 \end{bmatrix}$$

$$x = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}.$$

$$b = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-l}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 0 \end{bmatrix}.$$

There exists a unique solution for the c_i's.

The matrix is strictly diagonally dominant, so it is invertible and a unique solution for the c_i exists.

First we need a definition and a theorem.

Def. The $n \times n$ matrix A is strictily diagonally dominant if

$$|a_{ii}| > \sum_{j=1; j \neq i}^{n} |a_{ij}|.$$

holds for each $i = 1, \ldots, n$.

Thm. A strictly diagonally dominant matrix is invertible.

Also note that the matrix A is **tridiagonal**: all entires are zero except for a band which is 3 entries wide centred on the main diagonal.

Solutions to tridiagonal linear systems can be found very efficiently:

*Only O(n) operations are needed using methods we shall discuss later

Clamped Boundary Conditions We also want to treat clamped boundary conditions:

$$f'(a) = S'(a) = S_0'(x_0) = b_0 + 2c_0(x_0 - x_0) + 3d_0(x_0 - x_0)^2 = b_0.$$

but

$$b_{i-1} = \frac{a_i - a_{i-1}}{h_{i-1}} - \frac{h_{i-1}}{3}(2c_{i-1} + c_i).$$

$$\therefore b_0 = \frac{a_1 - a_0}{h_0} - \frac{h_0}{3}(2c_0 + c_1).$$

$$\implies 2h_0c_0 + h_0c_1 = \frac{3}{h_0}(a_1 - a_0) - 3f'(a).$$

Similarly,

$$h_{n-1}c_{n-1} + 2h_{n-1}c_n = 3f'(b) - \frac{3}{h_{n-1}}(a_{n-1} - a_n).$$

Once again, we obtain a strictly diagonally dominant linear system

$$\implies \text{A unique solution exists for the } c_i\text{'s}$$

Furthermore, the system is tridiagonal so it can be solved efficiently for the coefficients of the spline.