Numerical Analysis Report 1

谭希 TanXi[1]

[1] *Zhejiang University, Email: 3220100027@zju.edu.cn*

2025 年 1 月 1 日

Program Structure Description,Classes and Their Structures

- The `CubicBsplines` class is focused on B-spline interpolation
- The `pp_form` class provides the framework for polynomial interpolation, supporting linear and polynomial interpolation techniques and allowing for easy extension.

It mainly consists of two classes: `CubicBsplines` and `pp_form`. The `CubicBsplines` class is designed to perform cubic B-spline interpolation, while the `pp_form` class serves as an abstract base class for polynomial interpolation, allowing subclasses to implement various interpolation methods.

1 Class `CubicBsplines`

The `CubicBsplines` class is responsible for cubic B-spline interpolation, supporting two different boundary conditions: **Theorem 3.57's boundary conditions (complete boundary conditions)** and **Theorem 3.58's natural boundary conditions**.

- `computeCoeffs357` and `computeCoeffs358`: These methods compute the B-spline coefficients for different boundary conditions, relying on the Eigen library for matrix operations.
- `interpolate357` and `interpolate358`: These methods provide interfaces to perform interpolation using specific boundary conditions.
- `evaluate`: This method evaluates the value of the B-spline at a given point.
- `B3`: This is a static method that computes the value of the cubic B-spline basis function, supporting calculation for different segments of the B-spline.

2 Class `pp_form`

The `pp_form` class is an abstract class that defines common interfaces for polynomial interpolation. All specific polynomial interpolation classes, such as linear and cubic interpolation, should inherit from this class and implement its virtual functions. This class mainly manages the nodes and function values for interpolation and provides a framework for various polynomial interpolation techniques.

- `operator()`: Evaluates the polynomial at a given point x.
- `compute_Interpolation`: A pure virtual function that requires subclasses to implement the specific interpolation computation.
- `print_Expression`: A pure virtual function for printing the polynomial expression.

3 Classes `pp_form_linear` and `pp_form_cubic`

These two classes implement linear interpolation and cubic polynomial interpolation, respectively, and both inherit from the `pp_form` class.

3.0.1 `pp_form_linear`

- `compute_Interpolation`: Computes the coefficients for linear interpolation and fills the `coeffs` vector.
- `print_Expression`: Prints the linear polynomial expression for each interval.

3.1 `pp_form_cubic`

- `setup_basic_system`: Constructs and solves a linear system to compute the coefficients for cubic polynomials.
- `compute_Interpolation`: Implements the computation of cubic polynomial interpolation coefficients. It uses the Eigen library to set up and solve the linear system.

4 Class Relationship Explanation

- The `CubicBsplines` class focuses on cubic B-spline interpolation, providing methods to compute the spline coefficients and evaluate the spline at specific points. It operates independently and uses the Eigen library to perform matrix operations for solving linear systems.
- The `pp_form` class serves as the base class for polynomial interpolation. It defines a common interface for various polynomial interpolation methods. Subclasses can implement different interpolation techniques, such as linear and cubic interpolation.
- The `pp_form_linear` and `pp_form_cubic` classes inherit from `pp_form` and implement specific methods for linear and cubic polynomial interpolation, respectively.

Mathematical Theories Used

4.1 pp form splines

From Theorem 3.3, we have $m_i = s'(f; x_i)$, for $s \in \mathcal{S}_3^2$, for $i = 2, 3, \cdots, N-1$, there is:

$$\lambda_i m_{i-1} + 2m_i + \mu_i m_{i+1} = 3\mu_i f[x_i, x_{i+1}] + 3\lambda_i f[x_{i-1}, x_i] \tag{1}$$

In the formula,

$$\mu_i = \frac{x_i - x_{i-1}}{x_{i+1} - x_{i-1}} = \frac{x_i - x_{i-1}}{x_{i+1} - x_i + x_i - x_{i-1}} = \frac{h_i}{h_{i+1} + h_i}, \quad \lambda_i = \frac{x_{i+1} - x_i}{x_{i+1} - x_{i-1}} = \frac{h_{i+1}}{h_{i+1} + h_i}, \tag{2}$$

4.1.1 Cubic Splines

Cubic splines satisfy the boundary conditions

$$s'(f;a) = f'(a) \;\; and \;\; s'(f;b) = b \tag{3}$$

That is, the first-order derivatives of the curve at the two endpoints of the interval are given. Let $b_i = 3\mu_{i+1}f[x_{i+1}, x_{i+2}] + 3\lambda_{i+1}f[x_i, x_{i+1}]$, for $i = 1, \ldots, N-2$, then we have:

$$\begin{bmatrix} 2 & \mu_2 & & & & & \\ \lambda_3 & 2 & \mu_3 & & & & \\ & & \ddots & & & & \\ & & \lambda_i & 2 & \mu_i & & \\ & & & & \ddots & & \\ & & & & \lambda_{N-2} & 2 & \mu_{N-2} \\ & & & & & \lambda_{N-1} & 2 \end{bmatrix} \begin{bmatrix} m_2 \\ m_3 \\ \vdots \\ m_i \\ \vdots \\ m_{N-2} \\ m_{N-1} \end{bmatrix} = \mathrm{b} = \begin{bmatrix} b_1 - \lambda_2 m_1 \\ b_2 \\ \vdots \\ b_{i-1} \\ \vdots \\ b_{N-3} \\ b_{N-2} - \mu_{N-1} m_N \end{bmatrix} \tag{4}$$

4.1.2 Natural Cubic Splines

Natural cubic splines are known for the second-order derivative values at the two endpoints:

$$s''(f;a) = 0 \;\; s''(f;b) = 0 \tag{5}$$

From Theorem 3.3, we know

$$s_i(x) = f_i + (x - x_i)m_i + (x - x_i)^2 \frac{K_i - m_i}{x_{i+1} - x_i} + (x - x_i)^2(x - x_{i+1}) \frac{m_i + m_{i+1} - 2K_i}{(x_{i+1} - x_i)^2} \tag{6}$$

Taking the second derivative of the above formula, we get

$$s_i''(x) = 2\frac{K_i - m_i}{x_{i+1} - x_i} + 2(x - x_{i+1}) \frac{m_i + m_{i+1} - 2K_i}{(x_{i+1} - x_i)^2} + 4(x - x_i) \frac{m_i + m_{i+1} - 2K_i}{(x_{i+1} - x_i)^2} \tag{7}$$

Organizing, we get:

$$s_i''(x) = \frac{6x - 4x_i - 2x_{i+1} - 2h_i}{h_i^2} m_i + \frac{6x - 4x_i - 2x_{i+1}}{h_i^2} m_{i+1} + K_i \frac{-12x + 8x_i + 4x_{i+1} + 2h_i}{h_i^2} \tag{8}$$

The above formula can be transformed into:

$$s_i''(x) = \frac{6x - 2x_i - 4x_{i+1}}{h_i^2} m_i + \frac{6x - 4x_i - 2x_{i+1}}{h_i^2} m_{i+1} + \frac{-12x + 6x_i + 6x_{i+1}}{h_i^2} K_i \tag{9}$$

If the curve is on $[x_1, x_2]$, let $s_i''(x_1) = 0$, then we have:

$$s_1''(x_1) = -\frac{4}{h_1} m_0 - \frac{2}{h_1} m_1 + \frac{6}{h_1} \frac{f_2 - f_1}{h_1} \tag{10}$$

Then we have:

$$2m_0 + m_1 = 3f[x_1, x_2] - \frac{2}{h_1} s_1''(x_1) = 3f[x_1, x_2] \tag{11}$$

Similarly, we can get $m_{N-1} + 2m_N = 3f[x_{N-1}, x_N]$

So we have:

$$\begin{cases} 2m_1 + m_2 = 3f[x_1, x_2] = g_0 \\ m_{N-1} + 2m_N = 3f[x_{N-1}, x_N] = g_{N-1} \end{cases} \tag{12}$$

Thus, we get:

$$\begin{bmatrix} 2 & 1 & & & & & \\ \lambda_2 & 2 & \mu_2 & & & & \\ & & \ddots & & & & \\ & & \lambda_i & 2 & \mu_i & & \\ & & & & \ddots & & \\ & & & & \lambda_{N-1} & 2 & \mu_{N-1} \\ & & & & & 1 & 2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \vdots \\ m_i \\ \vdots \\ m_{N-1} \\ mN \end{bmatrix} = \mathrm{b} = \begin{bmatrix} g_0 \\ b_1 \\ \vdots \\ b_{i-1} \\ \vdots \\ b_{N-2} \\ g_{N-1} \end{bmatrix} \tag{13}$$

Among them

4.1.3 Periodic Cubic Splines

Periodic cubic splines satisfy:

$$s(f;b) = s(f;a), s'(f;b) = s'(f;a), s''(f;b) = s''(f;a) \tag{14}$$

That is

$$s(x_1) = s(x_N) \quad s'(x_1) = s'(x_N) \quad s''(x_1) = s''(x_N) \tag{15}$$

From the condition of equal first derivatives, we get:

$$s'_1(x_1) = m_1 \tag{16}$$

$$s'_{N-1}(x_N) = m_N \tag{17}$$

From the condition of equal second derivatives, we get:

$$-\frac{4}{h_1}m_1 - \frac{2}{h_1}m_2 + \frac{6}{h_1}K_1 = \frac{2}{h_{N-1}}m_{N-1} + \frac{4}{h_{N-1}}m_N - \frac{6}{h_{N-1}^2}K_{N-1} \tag{18}$$

Further derivation gives:

$$\frac{1}{h_1}m_2 + \frac{1}{h_{N-1}}m_{N-1} + (\frac{1}{h_{N-1}} + \frac{1}{h_1})m_N = 3\frac{1}{h_1}K_1 + 3\frac{1}{h_{N-1}}K_{N-1} \tag{19}$$

Let $\lambda_N = \frac{x_2 - x_1}{x_2 - x_1 + x_N - x_{N-1}}$ and $\mu_N = \frac{x_N - x_{N-1}}{x_2 - x_1 + x_N - x_{N-1}}$, then we have:

$$\begin{cases} m_1 = m_N \\ \mu_N m_2 + 2\lambda_N m_{N-1} + 2m_N = 3(\mu_N f[x_1, x_2] + \lambda_N f[x_{N-1}, x_N]) \end{cases} \tag{20}$$

Thus, we have the system of equations:

$$\begin{bmatrix} 2 & \mu_2 & & & & & & \\ \lambda_3 & 2 & \mu_3 & & & & & \\ & & \ddots & & & & & \\ & & \lambda_i & 2 & \mu_i & & & \\ & & & & \ddots & & & \\ & & & & \lambda_{N-2} & 2 & \mu_{N-2} & \\ & & & & & \lambda_{N-1} & 2 & \mu_{N-1} \\ \mu_N & & & & & & 2\lambda_N & 2 \end{bmatrix} \begin{bmatrix} m_2 \\ m_3 \\ \vdots \\ m_i \\ \vdots \\ m_{N-2} \\ m_{N-1} \\ m_N \end{bmatrix} = \mathrm{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_i \\ \vdots \\ b_{N-2} \\ g_N \end{bmatrix} \tag{21}$$

4.2 B-splines

From Theorem 3.57, there exists a unique $S(x) \in \mathcal{S}_3^2$ that interpolates $f(x)$ at points $1, 2, \cdots, N$, and has $S'(1) = f'(1)$ and $S'(N) = f'(N)$. Therefore, the B-spline interpolation function is:

$$S(x) = \sum_{i=-1}^{N} a_i B_i^3(x) \tag{22}$$

The coefficients a_i are obtained by solving the linear system $Ma = b$.

$$f(t_i) = S(t_i) = a_{i-2}B_{i-2}^3(t_i) + a_{i-1}B_{i-1}^3(t_i) + a_i B_i^3(t_i) \tag{23}$$

Based on the above equation and introducing boundary conditions, the coefficients $a = [a_{-1}, a_0, a_1, \cdots, a_N]^T$ can be solved.

4.2.1 Clamped Cubic Splines

For clamped cubic splines, there are boundary conditions where the first derivatives at the endpoints are equal:

$$S'(t_1) = f'(t_1) \tag{24}$$

$$S'(t_N) = S'(t_N) \tag{25}$$

And $S'(x) = \sum_{i=-1}^{N} a_i B_i'^3(x)$. Thus, the linear system is:

$$M = \begin{bmatrix} B_{-1}'^3(t_1) & B_0'^3(t_1) & B_1'^3(t_1) & & & \\ B_{-1}^3(t_1) & B_0^3(t_1) & B_1^3(t_1) & & & \\ & \ddots & \ddots & \ddots & & \\ & & B_{N-2}^3(t_N) & B_{N-1}^3(t_N) & B_N^3(t_N) \\ & & B_{N-2}'^3(t_N) & B_{N-1}'^3(t_N) & B_N'^3(t_N) \end{bmatrix} \tag{26}$$

$$b^T = [f'(t_1), f(t_1), \cdots, f(t_N), f'(t_N)] \tag{27}$$

4.2.2 Periodic Boundary Conditions

For periodic boundary conditions, we have:

$$S(t_1) = S(t_N) \tag{28}$$

$$S'(t_1) = S'(t_N) \tag{29}$$

$$S''(t_1) = S''(t_N) \tag{30}$$

Thus, we have:

$$M = \begin{bmatrix} -B_{-1}'^3(t_1) & -B_0'^3(t_1) & -B_1'^3(t_1) & 0 & \cdots & 0 & B_{N-2}'^3(t_N) & B_{N-1}'^3(t_N) & B_N'^3(t_N) \\ B_{-1}^3(t_1) & B_0^3(t_1) & B_1^3(t_1) & & & & & & \\ & \ddots & \ddots & \ddots & & & & & \\ & & & & & 0 & B_{N-2}^3(t_N) & B_{N-1}^3(t_N) & B_N^3(t_N) \\ -B_{-1}''^3(t_1) & -B_0''^3(t_1) & -B_1''^3(t_1) & 0 & \cdots & 0 & B_{N-2}''^3(t_N) & B_{N-1}''^3(t_N) & B_N''^3(t_N) \end{bmatrix} \tag{31}$$

$$b^T = [0, f(t_1), \cdots, f(t_{N-1}), f(t_N), 0]^T \tag{32}$$

4.2.3 Natural Splines

The natural boundary conditions are:

$$S''(t_1) = S''(t_N) = 0 \tag{33}$$

Thus, by solving the linear system $Ma = b$, the coefficients of the B-splines are obtained:

$$M = \begin{bmatrix} B''^3_{-1}(t_1) & B''^3_0(t_1) & B''^3_1(t_1) & & \\ B^3_{-1}(t_1) & B^3_0(t_1) & B^3_1(t_1) & & \\ & \ddots & \ddots & \ddots & \\ & & B^3_{N-2}(t_N) & B^3_{N-1}(t_N) & B^3_N(t_N) \\ & & B''^3_{N-2}(t_N) & B''^3_{N-1}(t_N) & B''^3_N(t_N) \end{bmatrix} \tag{34}$$

$$b^T = [0, f(t_1), \cdots, f(t_N), 0]^T \tag{35}$$