Numerical Methods

Splines

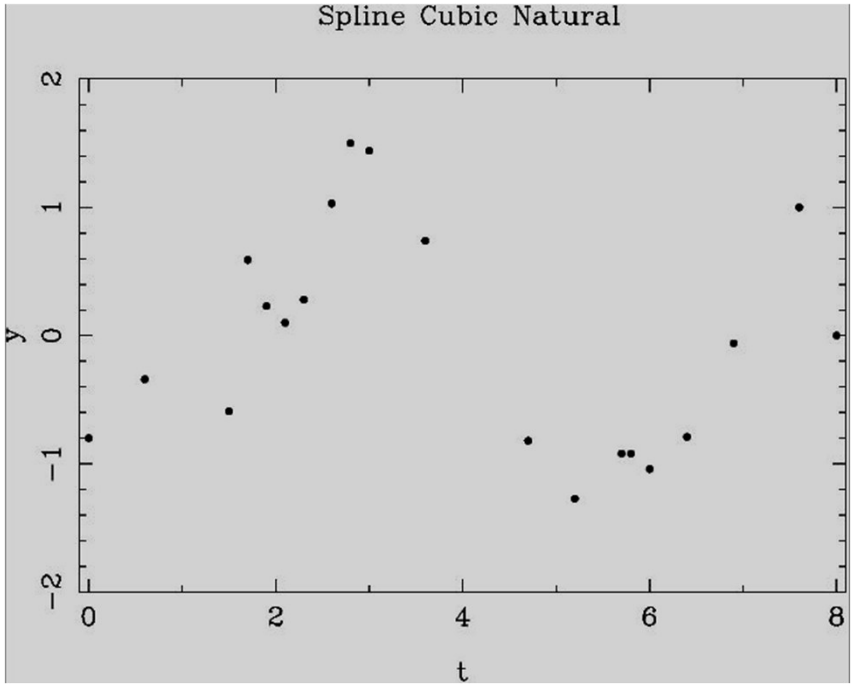
- Piecewise Interpolation

2022-2023

[DΣIM]

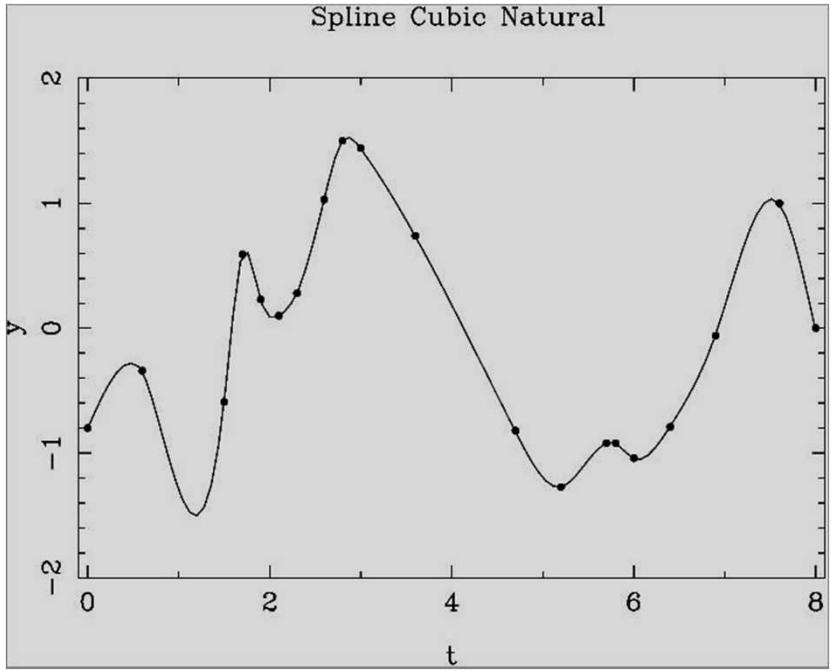
Splines

- Suppose that we want to recover the information contained in certain table of data points:



Splines

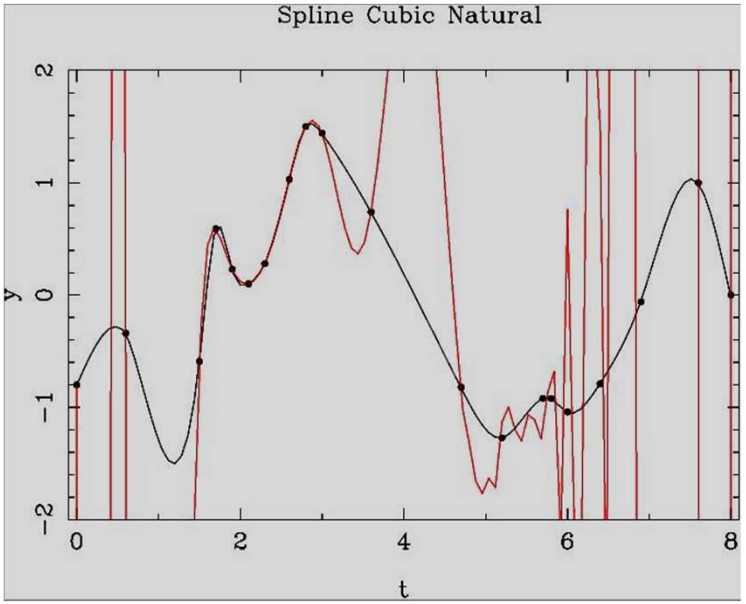
- Intuition suggest that the data should be represented by a function like this one:



[DΣIM]

Splines

- However, if we use the classical polynomial satisfying all the 20 points, we get something quite different:



[DΣIM]

Splines

- The error of the interpolation polynomial can be reduced only minimizing the product

$$(x-x_0)(x-x_1)\cdots(x-x_n)$$

- We can reduce this error using Chebyshev coordinates. We could think also of increase the number of nodes and reduce the distance $x - x_i$, but this will increase other factors in the error term.

Splines

- This suggests the idea of splitting the interval in smaller part and use less nodes in each subinterval. The interpolation polynomials build in this way have degrees lower or equal to 3 and are called ***splines***.
- Given an interval $[a, b]$ we introduce a partition:

$$\Delta = \{a = x_0 < x_1 < \cdots < x_n = b\}$$

- Where the values $x_i, i = 0,1, \ldots, n$ are called ***nodes***.

Splines

- In each interval $[x_i, x_{i+1}]$ we can introduce additional interpolation points and build an interpolating polynomial for this interval. Using polynomials of degree 3 we obtain an interpolation using ***cubic splines***.
- Given the partition $\Delta = \{x_0, \dots, x_n\}$ the cubic spline S_Δ associated to Δ and $[a, b]$ will be a ***piecewise polynomial*** that will join at the nodes with precise continuity conditions, namely the piecewise polynomials will be equal at the nodes up to the second derivative.

Splines

- Given the set of nodes, the spline functions are not easy to define explicitly, as they will have different expressions for each subinterval

$$S_\Delta(x) = \begin{cases} S_0(x) & x \in [x_0, x_1] \\ S_1(x) & x \in [x_1, x_2] \\ \vdots & \vdots \\ S_{n-1}(x) & x \in [x_{n-1}, x_n] \end{cases}$$

Splines

- Each of the functions $S_i(x)$ will be a polynomial. If we use first degree polynomials, i.e., straight lines we have a linear piecewise polynomial:

$$S_i(x) = a_i x + b_i$$

- But this will give us a continuous but broken line. We will see that is more efficient to use third degree polynomials or cubic splines:

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

Splines

- Given the set of nodes $x_0, x_1, \ldots, x_n$ and the value of the coefficients a_i, b_i, c_i, d_i, of each polynomial we evaluate the function, first determining in which interval our value x is located, and then using the corresponding $S_i(x)$ function
- If the function $S_\Delta(x)$ so defined is only continuous at the nodes we will call it ***spline function of first degree or linear splines***

Linear Splines

- A spline function of first degree, $S_\Delta(x)$is characterized by the following properties:
  - The domain of definition of $S_\Delta(x)$ is an interval $[a, b]$ of the real line
  - $S_\Delta(x)$ is continuous in this interval $[a, b]$
  - There is some partition of the interval:

$$\Delta = \{a = x_0 < x_1 < \cdots < x_n = b\}$$

  - So that, $S_\Delta(x)$ is a lineal polynomial in each interval $[x_i, x_{i+1}]$

Linear Splines

- Splines of first degree are useful when making linear interpolations. Suppose that we have a certain table of data points

x	x_1	x_2	x_3	...	x_n
y	y_1	y_2	y_3	...	y_n

- Which represents a set of points in the plane: (x_i,y_i)

Linear Splines

- For each pair of points (x_i, y_i) and (x_{i+1}, y_{i+1}) we can get a straight line of equation:

$$S_i(x) = y_i + m_i(x - x_i)$$

- Where m_i is the slope of the straight line, given by the expression:

$$m_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

Linear Splines

- This expression is useful when evaluating the function. The interval where the data point is located can be calculated by searching the first nonnegative, term in :

$$(x - x_{n-1}),\ (x - x_{n-2}),\ \ldots,\ (x - x_1)$$

- And then evaluate directly the linear expression

$$S_i(x) = y_i + m_i(x - x_i)$$

Quadratic Splines

- If we increase the degree of the polynomials, we can get smoother functions at the expense of longer computations.
- A spline function of second degree, $S_\Delta(x)$, or ***quadratic spline***, is a piecewise function defined using second degree polynomials with the condition that both $S_\Delta(x)$ and $S'_\Delta(x)$ must be continuous.
- We will see that this is no better than use cubic splines for each interval.

Quadratic Splines

- If we have a partition $\Delta = \{x_1, \ldots, x_{n+1}\}$ with n nodes, then there will be $n-1$ subintervals and $n-2$ internal nodes. As each spline is a quadratic function of the form:

$$S_i(x) = a_i x^2 + b_i x + c_i$$

- We will need a total of $3(n-1)$ coefficients, and we will need $3(n-1)$ conditions to determine these functions uniquely.

Quadratic Splines

- In the extremes of each subinterval $[x_i, x_{i+1}]$ we must fulfill the following conditions

$$S_i(x_i) = y_i, \quad S_i(x_{i+1}) = y_{i+1}$$

- As there are $n-1$ subintervals we have a grand total of $2(n-1)$ conditions. Unfortunately, the continuity of S_Δ does not add new conditions, as in each node x_{i+1}:

$$S_i(x_{i+1}) - S_{i+1}(x_{i+1}) = y_{i+1} - y_{i+1} = 0$$

- As this is just a linear combination of the former conditions

Quadratic Splines

- The continuity of the derivatives of the quadratic functions in the inner subintervals gives a total of $n-2$ new conditions.
- We have then $2(n-1)+n-2=3n-4$ conditions to define the necessary $3n-3$ unknowns.
- This last degree of liberty can be imposed in several ways to obtain different second-degree splines.

Quadratic Splines

- In the case of quadratic splines $Q(x)$, the piecewise function, defined as

$$Q(x) = \begin{cases} Q_1(x) & x_1 \le x \le x_2 \\ Q_2(x) & x_2 \le x \le x_3 \\ \vdots & \vdots \\ Q_n(x) & x_n \le x \le x_{n+1} \end{cases}$$

- Will be continuous and differentiable in the interval $[a, b]$ and will satisfy the data points, at the nodes, i.e., $Q(x_i) = y_i$

Quadratic Splines

- To build this function Q, we will need two sets of values $\{y_1, \dots, y_{n+1}\}$ and $\{z_1, \dots, z_{n+1}\}$ such that $Q(x_i) = y_i$ and for the derivatie, also $Q'(x_i) = z_i$ to completely define this quadratic function Q.
- In each subinterval, this second-degree polynomial can be written as:

$$Q_i(x) = \frac{z_{i+1} - z_i}{2(x_{i+1} - x_i)}(x - x_i)^2 + z_i(x - x_i) + y_i$$

Quadratic Splines

- If we impose now the continuity conditions at the nodes $Q_i(x_{i+1}) = Q_{i+1}(x_{i+1}), i = 2,3, \dots, n$ we obtain the relation:

$$z_{i+1} = -z_i + 2\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right), \quad 0 \le i \le n\text{-}1$$

- Giving a recursive relation to compute the z_i values from the data points and the additional constraint: $z_1 = Q'(x_1)$

Cubic Splines

- Splines of first and second degree are only useful to local interpolation as they share a common problem. Their second derivatives do not need to be continuous.
- This limits severely their interest for graphic representations. In the case of first-degree splines, we obtain broken lines, while for second degree splines as the curvature of the curve changes in each node giving a final shape not visually satisfactory

Cubic Splines

- We will need ***cubic polynomials***. The restriction of S_Δ to the subinterval $[x_i, x_{i+1}]$, $i = 1,2,\dots,n$, $S_{[x_i,x_{i+1}]}$ will be a polynomial of degree 3 at the most. $S_{[x_i,x_{i+1}]} \in P_3(\mathbb{R})$
- The cubic spline is a continuous function which has continuous second derivatives. This can be expressed as $S_\Delta \in C^2[a,b]$
- If S_Δ interpolates the function f, then

$$S_\Delta(x_i) = f(x_i), \qquad i = 1,2,\dots,n+1$$

Cubic Splines

- To build a cubic spline we will need to satisfy the conditions:

$$S_{\Delta}(x_i) = f(x_i), \qquad i = 1, 2, \ldots, n$$

- And the additional conditions of continuity:

$$S_{[x_{i-1},x_i]}(x_i) = S_{[x_i,x_{i+1}]}(x_i)$$
$$S'_{[x_{i-1},x_i]}(x_i) = S'_{[x_i,x_{i+1}]}(x_i) \qquad i = 2, 3, .., n-1$$
$$S''_{[x_{i-1},x_i]}(x_i) = S''_{[x_i,x_{i+1}]}(x_i)$$

- This gives a total of $(n) + 3(n-2) = 4n - 6$ constraints for a total of $4(n-1)$ unknowns to define the n pieces of the cubic spline.

Cubic Splines

- We will have always ***two degrees of freedom*** to build the cubic spline.
- Depending on how we select these two degrees of freedom, we can define alternative cubic splines.
- If the second derivatives at the ends of the interval are set to zero $S''_{\Delta}(a) = S''_{\Delta}(b) = 0$, we obtain the ***natural splines***.
- Choosing $S_{\Delta}^{(k)}(a) = S^{(k)}{}_{\Delta}(b)$, $k = 1,2$, we obtain ***periodic splines***.

Natural Splines

- ***Theorem.*** If f is defined at $a = x_1 < x_1 < \cdots < x_n = b$, then f has a unique natural spline interpolant S on the nodes $x_1, \ldots, x_n$ satisfying the boundary conditions $S''(a) = 0$ and $S''(b) = 0$.
- If we write the cubic polynomials as:

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \quad j = 1, 2, \ldots, n-1$$

- The interpolating conditions imply for the $n - 2$ intermediate nodes:

$$S_j(x_j) = a_j = f(x_j) \quad j = 2, \ldots, n-1$$

Natural Splines

- While

$$a_{j+1} = S_{j+1}(x_{j+1}) = S_j(x_{j+1}) = a_j + b_j(x_{j+1} - x_j) + c_j(x_{j+1} - x_j)^2 + d_j(x_{j+1} - x_j)^3$$
$$j = 1, \ldots, n-1$$

- As the terms $x_{j+1} - x_j$ will be used repeatedly it is convenient to introduce the notation:

$$h_j = x_{j+1} - x_j, \quad j = 1, 2, \ldots, n-1$$

- If we also define

$$a_n = S_n(x_n) = f(x_n), \quad j = 1, 2, \ldots, n$$

Natural Splines

- We obtain the equations:

$$a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3, \quad j = 1, \ldots, n-1$$

- In a similar manner, define $b_j = S'(x_j)$ then as:

$$S'_j(x) = b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2$$

- The continuity of the first derivatives imply

$$b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2, \quad j = 1, 2, \ldots, n$$

Natural Splines

- Similarly, if we define $c_j = S''(x_j)/2,\ i = 1,..,n$ we obtain from the continuity of the second derivatives, the relations:

$$c_{j+1} = c_j + 3d_j h_j \Leftrightarrow d_j = \frac{\left(c_{j+1} - c_j\right)}{3h_j}, \quad j = 1,...,n-1$$

- Now, solving for d_j we obtain for $j = 1, \ldots, n-1$:

$$a_{j+1} = a_j + b_j h_j + \frac{h_j^2}{3}\left(2c_j + c_{j+1}\right)$$

$$b_{j+1} = b_j + h_j\left(c_j + c_{j+1}\right)$$

Natural Splines

- Solving for b_j in the first equation

$$b_j = \frac{1}{h_j}\left(a_{j+1} - a_j\right) - \frac{h_j}{3}\left(2c_j + c_{j+1}\right), \quad j = 1,\ldots,n-1$$

- And reducing the order of j:

$$b_{j-1} = \frac{1}{h_{j-1}}\left(a_j - a_{j-1}\right) - \frac{h_{j-1}}{3}\left(2c_{j-1} + c_j\right)$$

Natural Splines

- These two equations, can be combined in the second equation, again reducing the order of j to obtain:

$$h_{j-1}c_{j-1} + 2\left(h_{j-1} + h_j\right)c_j + h_j c_{j+1} = \frac{3}{h_j}\left(a_{j+1} - a_j\right) - \frac{3}{h_{j-1}}\left(a_j - a_{j-1}\right)$$

$$j = 2, \ldots, n-1$$

- These equations involve only the $c_j, j = 2, \ldots, n-1$ as unknowns as the values $h_j, j = 1, \ldots, n-1$ and $a_j, j = 1, \ldots, n+1$ are given by the spacing of the nodes and the values of f at the nodes.

Natural Splines

- If we can solve for the $c_j, j = 2, \ldots, n-1$ and then, with the conditions $c_1 = c_n = 0,$ we can compute the coefficients b_j and $d_j, j = 1, \ldots, n-1$ from the intermediate equations.
- Now if we set the conditions $S''(a) = 0 = S''(b)$ we can write a linear system of equations for the coefficients:
- $$\mathbf{Ax = b}$$
- Where, **A** is a $n \times n$ matrix, as follows:

[DΣIM]

Natural Splines

- With matrix:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ h_1 & 2(h_1+h_2) & h_2 & \ddots & \ddots & \vdots \\ 0 & h_2 & 2(h_2+h_3) & h_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & h_{n-1} & 2(h_{n-1}+h_n) & h_n \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}$$

Natural Splines

- And **b** and **x** the vectors:

$$\mathbf{b} = \begin{bmatrix} 0 \\ \frac{3}{h_2}(a_3 - a_2) - \frac{3}{h_1}(a_2 - a_1) \\ \vdots \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 0 \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix}$$

- Finally, as the system matrix is strictly diagonal dominant, it will be non-singular and the solution of the system, unique.

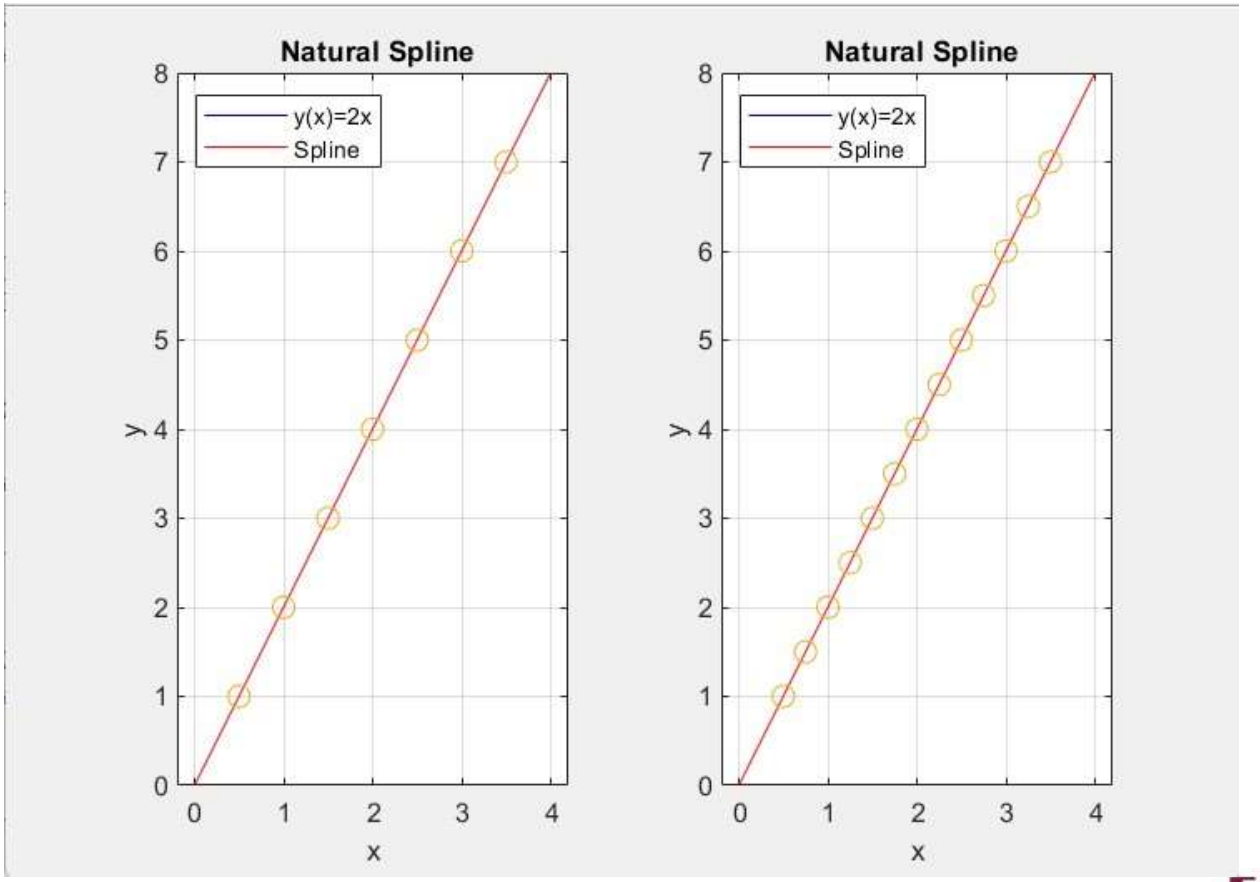
Natural Splines

- In the case of natural splines as $c_1 = 0 = c_n$ we can reduce the system to an order $(n-2) \times (n-2)$ with the matrices:

$$\mathbf{A} = \begin{pmatrix} 2(h_1+h_2) & h_2 & 0 & \cdots & 0 \\ h_2 & 2(h_2+h_3) & h_3 & \ddots & \vdots \\ 0 & h_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & h_{n-2} \\ 0 & \cdots & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix}$$

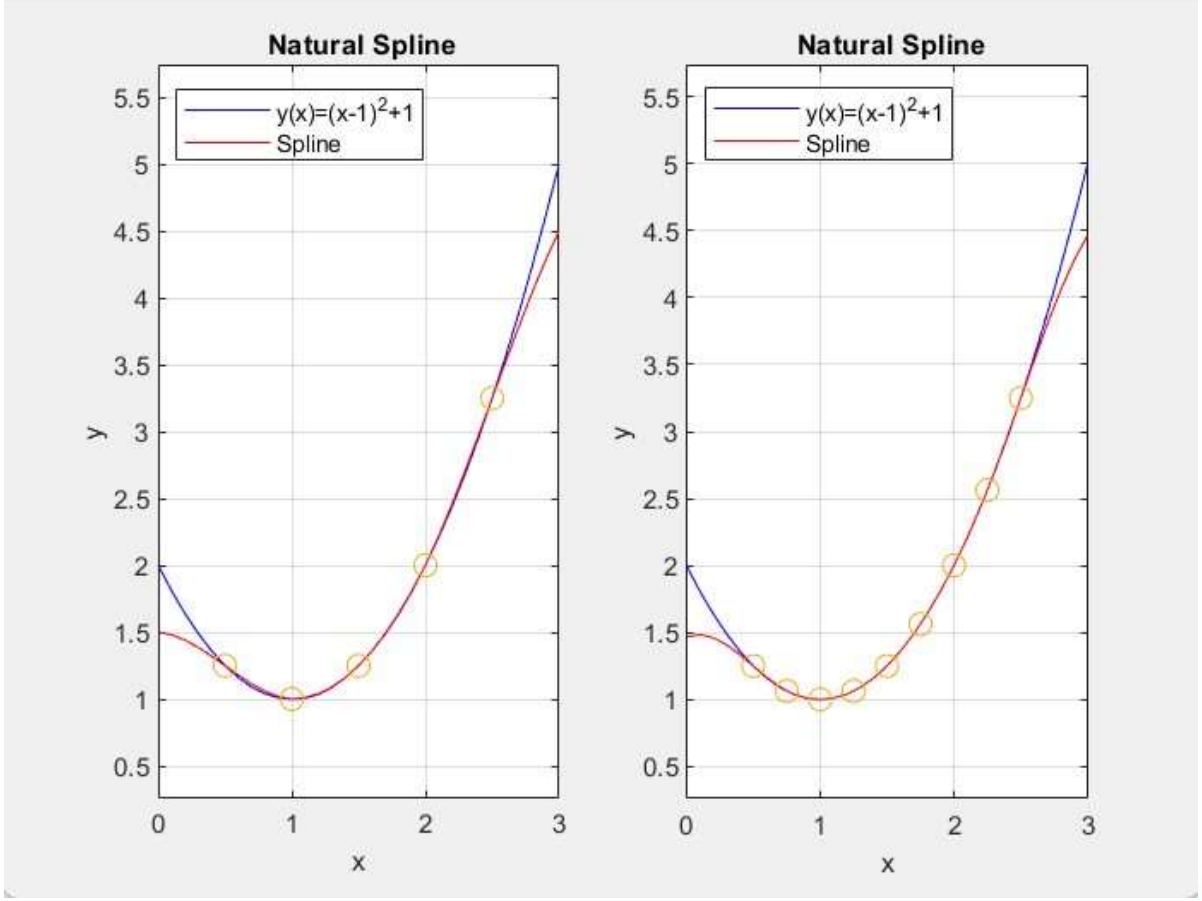
$$\mathbf{b} = \begin{bmatrix} \frac{3}{h_2}(a_3-a_2) - \frac{3}{h_1}(a_2-a_1) \\ \vdots \\ \vdots \\ \frac{3}{h_{n-1}}(a_n-a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1}-a_{n-2}) \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} c_2 \\ \vdots \\ \vdots \\ c_{n-1} \end{bmatrix}$$

Natural Spline $y = 2x$



[DEIM]

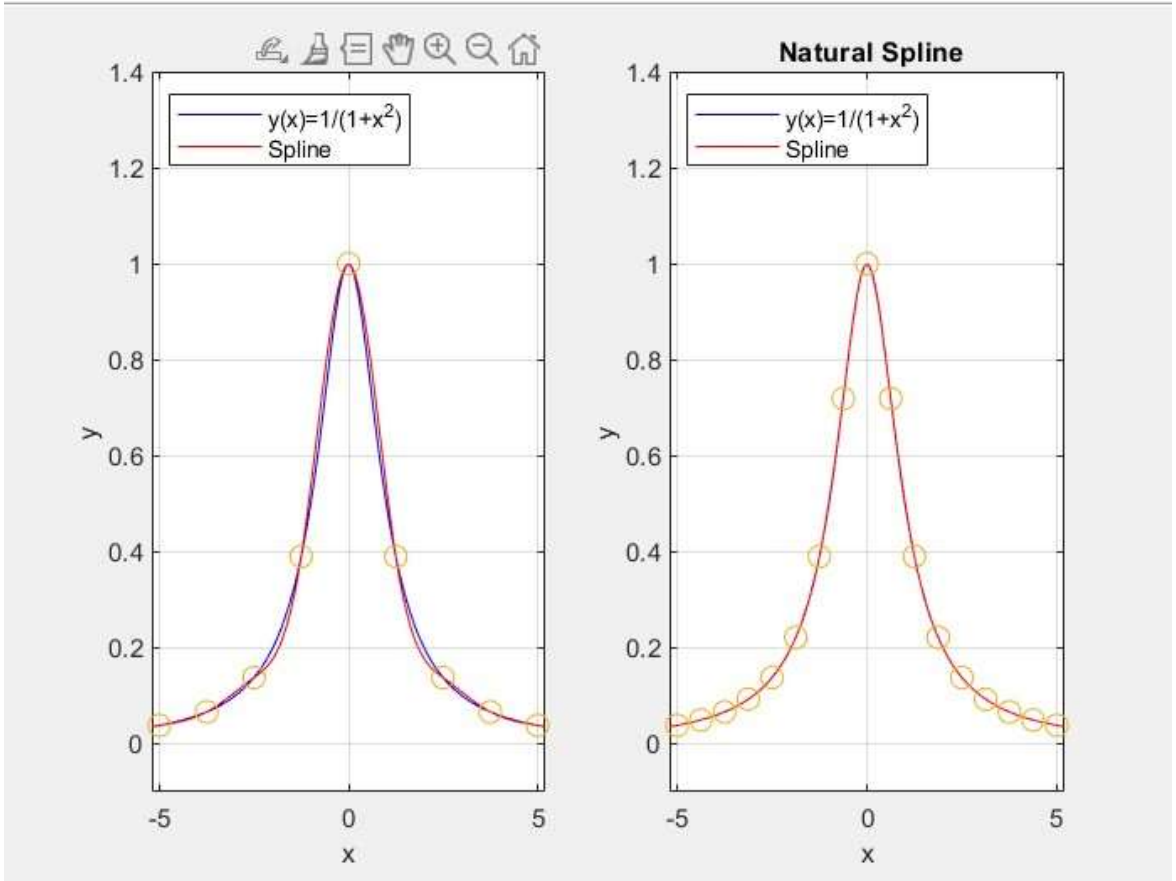
Natural Spline $y = (x-1)^2 + 1$



[DEIM]

Numerical Methods

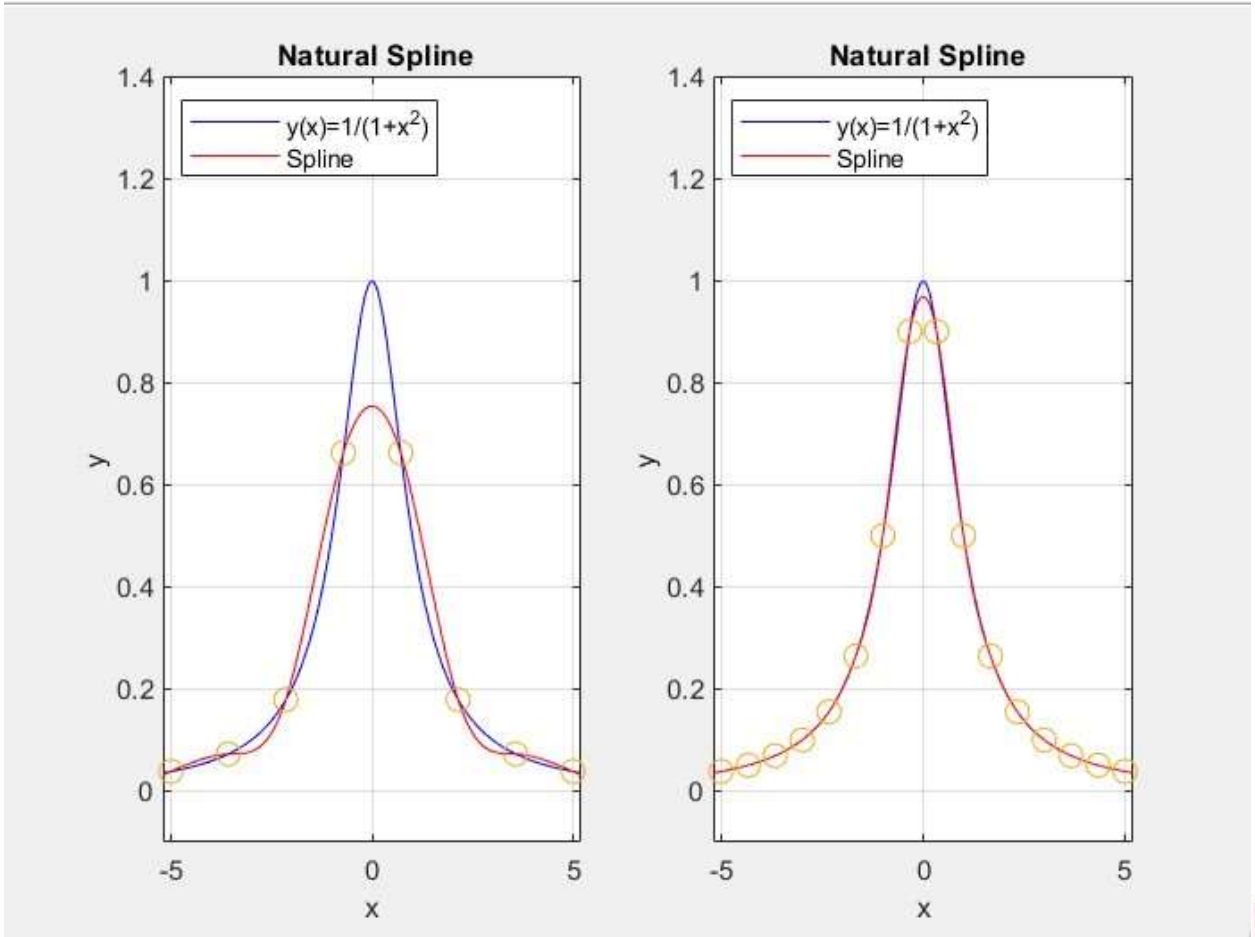
Natural Spline $y = 1/(1 + x^2)$



2022-2023

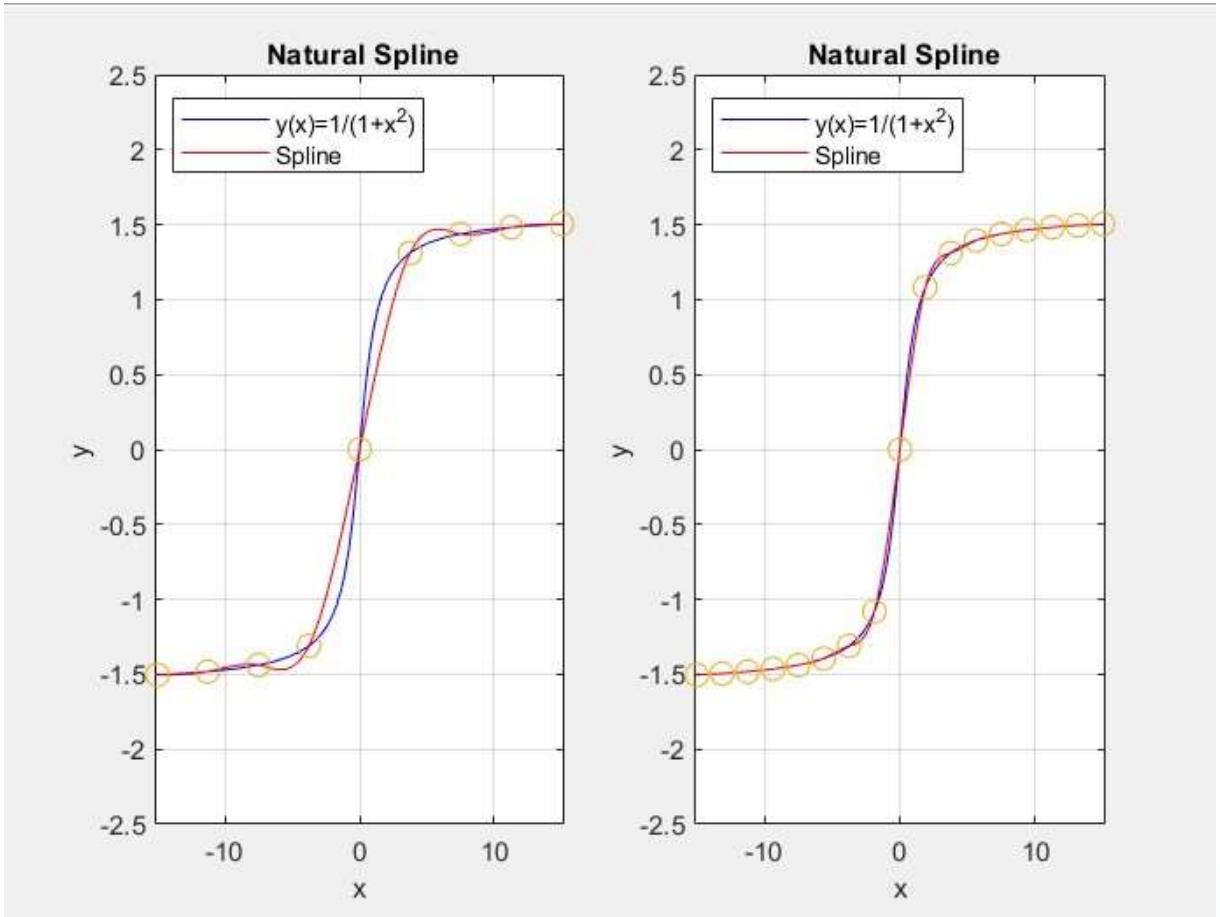
[DEIM]

Natural Spline $y = 1/(1 + x^2)$



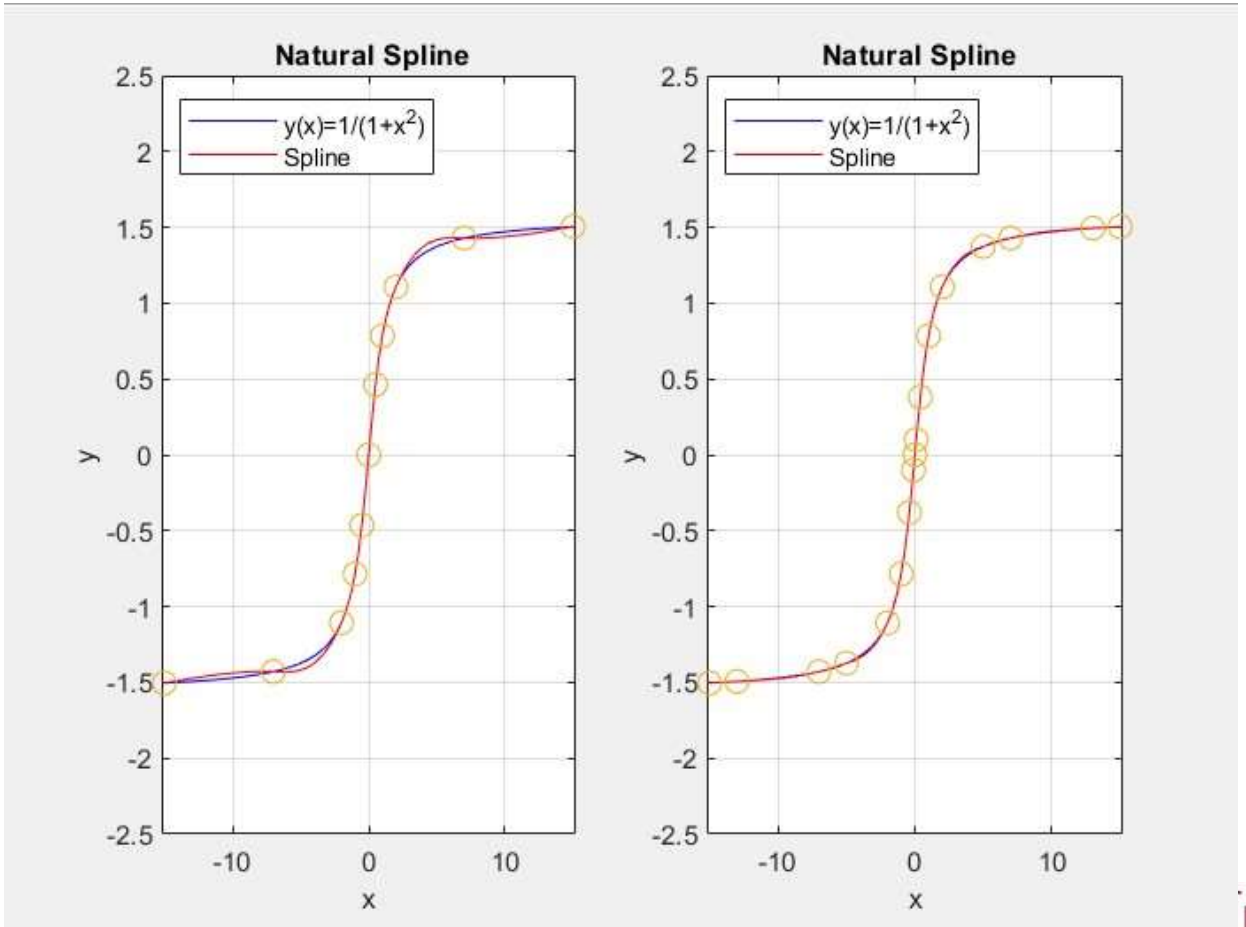
[DSIM]

Natural Spline $y = \arctan(x)$



[DΣIM]

Natural Spline $y = \arctan(x)$



Clamped Splines

- ***Theorem.*** If f is defined at $a = x_1 < x_1 < \cdots < x_n = b$, then f has a unique natural spline interpolant S on the nodes $x_1, \ldots, x_n$ satisfying the two boundary conditions $S'(a) = f'(a)$ and $S'(b) = f'(b)$.
- The reasoning in this case is like the case of natural splines and we can use the same set of equations. In this case, however, we must change some boundary conditions. As $f'(a) = S'(a) = S'(x_1) = b_1$, we obtain

$$f'(a) = \frac{1}{h_1}(a_2 - a_1) - \frac{h_1}{3}(2c_1 + c_2).$$

Clamped Splines

- From this equation we obtain:

$$2h_1c_1 + h_1c_2 = \frac{3}{h_1}\left(a_2 - a_1\right) - 3f'(a)$$

- Similarly, from:

$$f'(b) = b_n = b_{n-1} + h_{n-1}\left(c_{n-1} + c_n\right)$$

- We get:

- And:

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

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

Clamped Splines

- The system matrix is now $n \times n$:

$$\mathbf{A} = \begin{pmatrix} 2h_1 & h_1 & 0 & \cdots & \cdots & 0 \\ h_1 & 2(h_1+h_2) & h_2 & \ddots & \ddots & \vdots \\ 0 & h_2 & 2(h_2+h_3) & h_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & \cdots & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{pmatrix}$$

Clamped Splines

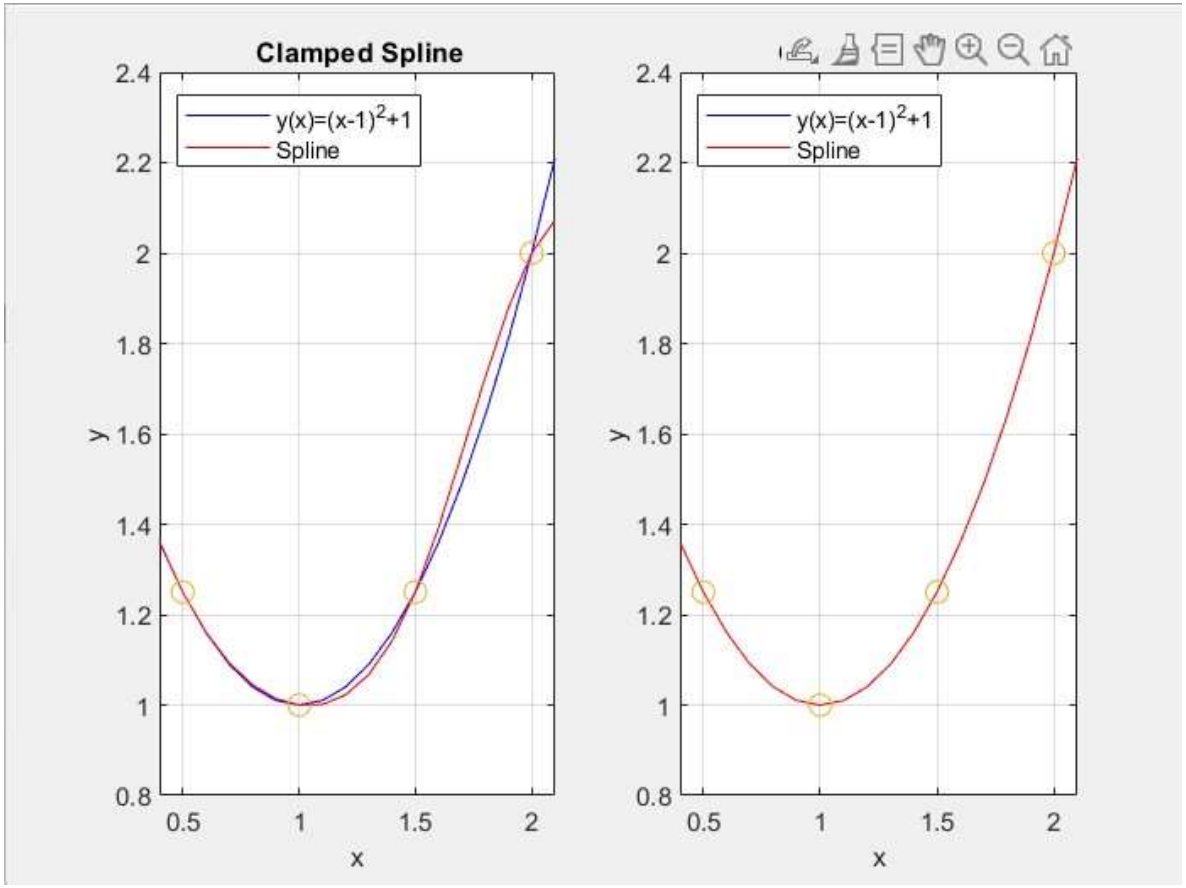
- While the vectors **b** and **x** are:

$$\mathbf{b} = \begin{bmatrix} \frac{3}{h_1}(a_2 - a_1) - 3f'(a) \\ \frac{3}{h_2}(a_3 - a_2) - \frac{3}{h_1}(a_2 - a_1) \\ \vdots \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 3f'(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix}$$

- Again, the system matrix is strictly diagonal dominant, and non-singular and the solution of the system will be unique.

Numerical Methods

Clamped Spline $y = (x-1)^2 + 1$



2022-2023

[DΣIM]