

Lecture 10 - B

§6.1, 6.2 Constructing Quadratic and Cubic Splines

T. Takahashi

Department of Mathematics and Statistics
University of New Mexico



We will answer these questions:

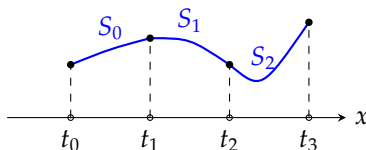
- How do you construct a quadratic spline?
- What is a cubic spline?
- Is the cubic spline unique?
- How do you construct a natural cubic spline?



Spline Interpolation

Method: Spline Interpolation

- **Given:** Set of points (t_i, y_i) , Degree of smoothness d
 - **Output:** Continuous piecewise polynomial $S(x)$
 - **Idea:** Splines interpolate data and do not suffer from large fluctuations between data points. The user controls the smoothness of the interpolation.
-
- User controls degree of smoothness
 - ▶ $d = 1$: connect the dots
 - ▶ $d = 2$: piecewise parabolas
 - ▶ $d = 3$: piecewise cubics



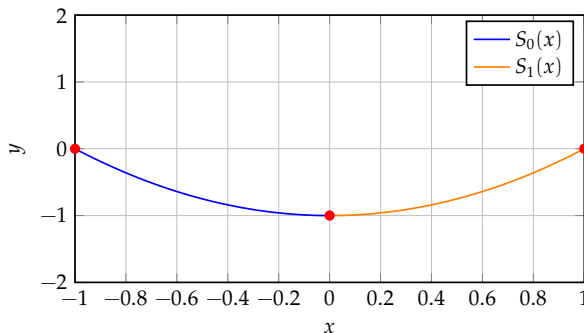
- Example: $d = 2$



Constructing Degree 2 Splines



Constructing a Degree 2 Spline



- Constructing a spline means finding all **6** unknown coefficients in

$$S_0(x) = ax^2 + bx + c$$

$$S_1(x) = dx^2 + ex + f$$

- We may set up a linear system



Six Conditions

Spline Formula

$$S_0(x) = ax^2 + bx + c$$

$$S_1(x) = dx^2 + ex + f$$

Conditions

- ① $S_0(t_0) = y_0$
- ② $S_0(t_1) = y_1$
- ③ $S_1(t_1) = y_1$
- ④ $S_1(t_2) = y_2$
- ⑤ $S'_0(t_1) = S'_1(t_1)$
- ⑥ $S'_0(t_0) = 0$ (extra condition)



Apply the Conditions

- Nodes:

$$(-1, 0) \quad (0, -1), \quad (1, 0)$$

- Interpolation at endpoints

$$S_0(-1) = 0 \implies a - b + c = 0$$

$$S_0(0) = -1 \implies c = -1$$

$$S_1(0) = -1 \implies f = -1$$

$$S_1(1) = 0 \implies d + e + f = 0$$

- Smoothness

$$S_0'(0) = S_1'(0) \implies b = e$$

- Extra condition

$$S_0'(-1) = 0 \implies -2a + b = 0$$



Linear System

- System

$$\begin{cases} a - b + c = 0 \\ c = -1 \\ f = -1 \\ d + e + f = 0 \\ b - e = 0 \\ -2a + b = 0 \end{cases}$$

- We may write this as

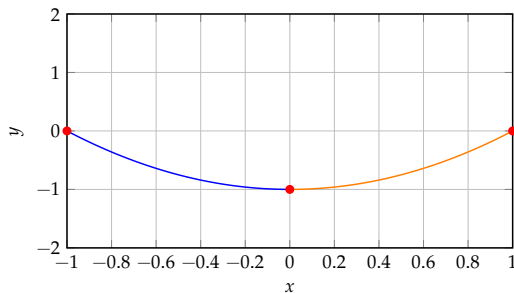
$$\begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ -2 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- The solution is

$$x = [1 \quad 0 \quad -1 \quad 1 \quad 0 \quad -1]^T$$



Degree 2 Spline



- Since the solution is

$$x = [1 \quad 0 \quad -1 \quad 1 \quad 0 \quad -1]^T$$

- The quadratic spline is given by

$$\begin{cases} x^2 - 1, & x \in [-1, 0] \\ x^2 - 1, & x \in [0, 1] \end{cases}$$



Bidiagonal System



Few Problems with the System

$$\begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ b_0 \\ c_0 \\ a_1 \\ b_1 \\ c_1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- Setting up this system by hand is very awkward
- Sparsity pattern is weird
- Size is $3n$

Key Idea

Solving for the coefficients directly is not scalable or structured.



A Better Idea

Instead of solving for coefficients, we change our unknowns.

New unknowns

Let

$$z_i := S'(t_i), \quad i = 0, 1, \dots, n$$

be the slope of the spline at each node.

- These values describe the **shape** of the spline
- The slopes at the nodes can derive the spline pieces $S_i(x)$

Key Idea

Instead of coefficients, we solve for the slopes at the nodes.



Reconstructing Each Piece

On each interval $[t_i, t_{i+1}]$, we want:

- $S_i(t_i) = y_i$
- $S_i(t_{i+1}) = y_{i+1}$
- $S'_i(t_i) = z_i$

A quadratic polynomial is uniquely determined by:

value + slope at two points

Key Idea

Knowing $y_i, y_{i+1}, z_i, z_{i+1}$ determines $S_i(x)$ completely.



Mini Example

Example

Find a quadratic polynomial of the form

$$p_2(x) = a(x - 1)^2 + b(x - 1) + c$$

such that:

- $p_2(1) = 2$
- $p_2(2) = 5$
- $p_2'(1) = 1$



Mini Example

Example

Find a quadratic polynomial of the form

$$p_2(x) = a(x - 1)^2 + b(x - 1) + c$$

such that:

- $p_2(1) = 2$
- $p_2(2) = 5$
- $p_2'(1) = 1$

Why this form?

We center the polynomial at $x = 1$ so the conditions at $x = 1$ become very simple.



Solving the Example

- First compute the derivative:

$$p_2'(x) = 2a(x - 1) + b$$

- Apply conditions at $x = 1$:

$$p_2(1) = c = 2$$

$$p_2'(1) = b = 1$$

- Now use conditions at $x = 2$:

$$p_2(2) = a(1)^2 + b(1) + c = a + b + c = 5$$

$$a + 1 + 2 = 5 \Rightarrow a = 2$$

- Result

$$p(x) = 2(x - 1)^2 + (x - 1) + 2$$



Constructing $S_i(x)$

- Instead of $S_i(x) = a_ix^2 + b_ix + c_i$, we will let

$$S_i(x) = \frac{z_{i+1} - z_i}{2h_i}(x - t_i)^2 + z_i(x - t_i) + y_i, \quad h_i = t_{i+1} - t_i$$

- The derivative is

$$S'_i(x) = \frac{z_{i+1} - z_i}{h_i}(x - t_i) + z_i$$

- Notice that we immediately get

- $S(t_i) = y_i$,
- $S'(t_i) = z_i$,
- $S'(t_{i+1}) = z_{i+1}$.



Equations for z_i

- Since $S(x)$ interpolates (t_{i+1}, y_{i+1}) we must have

$$S_i(t_{i+1}) = y_{i+1}$$

- Substituting $x = t_{i+1}$ into

$$\begin{aligned} S_i(t_{i+1}) &= \frac{z_{i+1} - z_i}{2h_i} (t_{i+1} - t_i)^2 + z_i(t_{i+1} - t_i) + y_i \\ &= \frac{z_{i+1} - z_i}{2} h_i + z_i h_i + y_i = y_{i+1} \end{aligned}$$

- With a bit of rearrangement

$$z_i + z_{i+1} = 2 \frac{y_{i+1} - y_i}{h_i}, \quad i = 0, \dots, n-1.$$

Question

What type of linear system will this equation yield?

The Bidiagonal System

Explicitly writing out this equation for values of $i \dots$

$$z_i + z_{i+1} = 2 \frac{y_{i+1} - y_i}{h_i} = r_i, \quad i = 0, \dots, n-1,$$

$$i = 0: \quad z_0 + z_1 = r_0$$

$$i = 1: \quad z_1 + z_2 = r_1$$

$$\vdots$$

$$i = n-1: \quad z_{n-1} + z_n = r_{n-1}$$



The Bidiagonal System

Explicitly writing out this equation for values of $i \dots$

$$z_i + z_{i+1} = 2 \frac{y_{i+1} - y_i}{h_i} = r_i, \quad i = 0, \dots, n-1,$$

$$i = 0: \quad z_0 + z_1 = r_0$$

$$i = 1: \quad z_1 + z_2 = r_1$$

$$\vdots$$

$$i = n-1: \quad z_{n-1} + z_n = r_{n-1}$$

In $Ax = b$ form, we get a **bidiagonal matrix**!

$$\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & \ddots & \ddots \\ & & & & & 1 & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ \vdots \\ r_{n-1} \end{bmatrix}$$



Quadratic Spline Algorithm

- 1 Choose an initial derivative value $z_0 = S'(t_0)$.
- 2 Use the recurrence to set up a linear system

$$z_i + z_{i+1} = 2 \frac{y_{i+1} - y_i}{t_{i+1} - t_i} =: r_i$$

$$\begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ & & \ddots & \ddots & \\ & & & 1 & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ \vdots \\ r_n \end{bmatrix}$$

- 3 Do back substitution and then compute the spline function with the formula

$$S_i(x) = \frac{z_{i+1} - z_i}{2h_i}(x - t_i)^2 + z_i(x - t_i) + y_i, \quad h_i = t_{i+1} - t_i$$



Summary

- A system in terms of the spline coefficients is awkward

$$\begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ b_0 \\ c_0 \\ a_1 \\ b_1 \\ c_1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- A system in terms of $S'(t_i) = z_i$ is structured

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ r_2 \end{bmatrix}$$

Key Idea

Advantages:

- Systematic
- Smaller linear system
- Fast $\mathcal{O}(n)$ solve



Cubic Spline



Third-Degree Spline or Cubic Spline

Definition: Degree Three Spline

A function $S(x)$ is a **spline of degree 3** if:

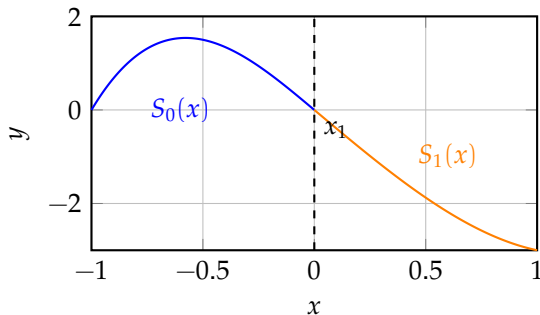
- ① The domain of $S(x)$ is an interval $[a, b]$
- ② $S(x)$ is continuous on $[a, b]$
- ③ $S'(x)$ is continuous on $[a, b]$
- ④ $S''(x)$ is continuous on $[a, b]$
- ⑤ There is a partition $a = t_0 < t_1 < \cdots < t_n = b$ such that $S(x)$ is cubic on each subinterval $[t_i, t_{i+1}]$.

Two differences from **degree two** splines:

- Condition 4: $S''(x)$ is continuous
- Condition 5: Each spline piece is cubic



Smoothness Condition



- For continuity of $S'(x)$ we need

$$S'_0(x_1) = S'_1(x_1)$$

- For continuity of $S''(x)$ we need

$$S''_0(x_1) = S''_1(x_1)$$



Number of Unknowns

Question

If there are $n + 1$ data points, how many unknowns does a cubic spline have?

- Each component of $S(x)$ has the form

$$S_i(x) = ax^3 + bx^2 + cx + d$$

- Cubics have **4 unknown coefficients** per subinterval
- $n + 1$ data points $\implies n$ subintervals

Number of Unknown Coefficients

A cubic spline has

$4n$ total unknowns



Number of Constraints

- Interpolation at the endpoints $S_i(t_i) = y_i$ and $S_i(t_{i+1}) = y_{i+1}$ so

2n constraints

- We need continuous 1st order derivative $S'_i = S'_{i+1}$ at interior nodes so

n-1 constraints

- We need continuous 2nd order derivative $S''_i = S''_{i+1}$ at interior nodes so

n-1 constraints

Number of Unknown Coefficients

A general cubic spline with $n + 1$ nodes has

4n-2 total constraints

Uniqueness for Cubic Splines

Underdetermined

$4n$ unknowns *vs.* **$4n-2$ constraints**

Common options to fill two missing constraints:

- **Natural:** $S''(t_0) = S''(t_n) = 0$
- **Clamped:** $S'(t_0) = a, S'(t_n) = b$
- **Periodic:** S' and S'' are periodic at the ends
 $S'(t_0) = S'(t_n)$ and $S''(t_0) = S''(t_n)$

Key Idea

- A generic cubic spline is **NOT** unique
- Cubic splines are **missing two constraints**
- A common choice to use include natural conditions

Tridiagonal System



Goal

Problem

- Given $n + 1$ data points (t_i, y_i) for $i = 0$ to n
- Assume natural conditions $S''(t_0) = S''(t_n) = 0$
- Construct the cubic spline

The Idea

- Instead of $S_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$, let

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + C_i(x - t_i) + D_i(t_{i+1} - x),$$

- Set up a linear system for $z_i = S''(t_i)$



Natural Cubic Spline: Step 1

- Notice the formula

$$S_i(x) = \underbrace{\frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3}_{\text{two cubics}} + \underbrace{C_i(x - t_i) + D_i(t_{i+1} - x)}_{\text{two linears}}$$

- First derivative

$$S'_i(x) = -\frac{z_i}{2h_i}(t_{i+1} - x)^2 + \frac{z_{i+1}}{2h_i}(x - t_i)^2 + C_i - D_i$$

- Second derivative

$$S''_i(x) = \frac{z_i}{h_i}(t_{i+1} - x) + \frac{z_{i+1}}{h_i}(x - t_i)$$



Natural Cubic Spline: Step 2

- We require these six conditions on the spline

- 1 $S_i(t_i) = y_i$
- 2 $S_i(t_{i+1}) = y_{i+1}$
- 3 $S'_i(t_i) = S'_{i-1}(t_i)$
- 4 $S'_i(t_{i+1}) = S'_{i+1}(t_{i+1})$
- 5 $S''_i(t_i) = z_i$
- 6 $S''_i(t_{i+1}) = z_{i+1}$

- Since $S''_i(x) = \frac{z_i}{h_i}(t_{i+1} - x) + \frac{z_{i+1}}{h_i}(x - t_i)$ we immediately have

- ▶ Condition 5: $S''_i(t_i) = z_i$
- ▶ Condition 6: $S''_i(t_{i+1}) = z_{i+1}$

- Now we enforce the other 4 conditions



Natural Cubic Spline: Step 3

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + C_i(x - t_i) + D_i(t_{i+1} - x)$$

- Apply Condition 1: $S_i(t_i) = y_i$ to get a formula for D_i

$$y_i = S_i(t_i) = \frac{z_i}{6h_i}(t_{i+1} - t_i)^3 + D_i(t_{i+1} - t_i)$$

$$= \frac{z_i}{6}h_i^2 + D_i h_i$$

$$\therefore D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i$$

- Apply Condition 2: $S_i(t_{i+1}) = y_{i+1}$ to get a formula for C_i

$$y_{i+1} = S_i(t_{i+1}) = \frac{z_{i+1}}{6h_i}(t_{i+1} - t_i)^3 + C_i(t_{i+1} - t_i)$$

$$= \frac{z_{i+1}}{6}(h_i)^2 + C_i h_i$$

$$\therefore C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$$



Natural Cubic Spline: Step 3

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1}-x)^3 + \frac{z_{i+1}}{6h_i}(x-t_i)^3 + \left(\frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}\right)(x-t_i) + \left(\frac{y_i}{h_i} - \frac{h_i}{6}z_i\right)(t_{i+1}-x)$$

- We know the derivatives

$$S'_i(x) = -\frac{z_i}{2h_i}(t_{i+1}-x)^2 + \frac{z_{i+1}}{2h_i}(x-t_i)^2 + C_i - D_i$$

$$S'_{i-1}(x) = -\frac{z_{i-1}}{2h_{i-1}}(t_i-x)^2 + \frac{z_i}{2h_{i-1}}(x-t_{i-1})^2 + C_{i-1} - D_{i-1}$$

- Apply Condition 3: $S'_i(t_i) = S'_{i-1}(t_i)$:

$$S'_i(t_i) = -\frac{h_i}{6}z_{i+1} + C_i - D_i$$

$$S'_{i-1}(t_i) = \frac{h_{i-1}}{6}z_i + C_{i-1} - D_{i-1}$$

- We have

$$-\frac{h_i}{6}z_{i+1} + C_i - D_i = \frac{h_{i-1}}{6}z_i + C_{i-1} - D_{i-1}$$



Natural Cubic Spline: Step 4

- Substitute C_i, D_i and simplify each side
- Left-hand side:

$$\begin{aligned} -\frac{h_i}{6}z_{i+1} + C_i - D_i &= -\frac{h_i}{6}z_{i+1} + \left(\frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}\right) - \left(\frac{y_i}{h_i} - \frac{h_i}{6}z_i\right) \\ &= \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}z_{i+1} + \frac{h_i}{6}z_i \end{aligned}$$

- Right-hand side:

$$\begin{aligned} \frac{h_{i-1}}{6}z_i + C_{i-1} - D_{i-1} &= \frac{h_{i-1}}{6}z_i + \left(\frac{y_i}{h_{i-1}} - \frac{h_{i-1}}{6}z_i\right) - \left(\frac{y_{i-1}}{h_{i-1}} - \frac{h_{i-1}}{6}z_{i-1}\right) \\ &= \frac{y_i - y_{i-1}}{h_{i-1}} + \frac{h_{i-1}}{6}z_{i-1} - \frac{h_{i-1}}{3}z_i \end{aligned}$$



Natural Cubic Spline: Step 5

- The simplified form of $S'_i(t_i) = S'_{i-1}(t_i)$ is:

$$\frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}z_{i+1} + \frac{h_i}{6}z_i = \frac{y_i - y_{i-1}}{h_{i-1}} + \frac{h_{i-1}}{6}z_{i-1} - \frac{h_{i-1}}{3}z_i$$

- Rearranging gives the final equation:

Formula for z 's

$$h_{i-1}z_{i-1} + 2(h_i + h_{i-1})z_i + h_i z_{i+1} = 6(b_i - b_{i-1}),$$

$$b_i = \frac{y_{i+1} - y_i}{h_i}$$



Tridiagonal System

- This is a recurrence equation

$$h_{i-1}z_{i-1} + 2(h_i + h_{i-1})z_i + h_i z_{i+1} = 6(b_i - b_{i-1}), \quad i = 1, \dots, n-1,$$

- Substituting for a few values of $i \dots$

$$i = 1: \quad h_0 z_0 + 2(h_1 + h_0)z_1 + h_1 z_2 = 6(b_1 - b_0)$$

$$i = 2: \quad h_1 z_1 + 2(h_2 + h_1)z_2 + h_2 z_3 = 6(b_2 - b_1)$$

$$i = 3: \quad h_2 z_2 + 2(h_3 + h_2)z_3 + h_3 z_4 = 6(b_3 - b_2)$$

$$\vdots$$

$$i = n-1: \quad h_{n-2}z_{n-2} + 2(h_{n-1} + h_{n-2})z_{n-1} + h_{n-1}z_n = 6(b_{n-1} - b_{n-2})$$

Observation

Each equation involves only three neighboring unknowns:

$$z_{i-1}, \quad z_i, \quad z_{i+1}.$$

Natural Cubic Spline: Summary

The z_i are defined by

$$h_{i-1}z_{i-1} + 2(h_i + h_{i-1})z_i + h_i z_{i+1} = 6(b_i - b_{i-1}) =: v_i$$

- For natural conditions $z_0 = z_n = 0$, from $S''(t_0) = S''(t_n) = 0$
- Set up a **tridiagonal linear system** and solve it in $\mathcal{O}(n)$ FLOPs

$$\begin{bmatrix} 1 & & & & & & & & \\ h_0 & u_1 & h_1 & & & & & & \\ & h_1 & u_2 & h_2 & & & & & \\ & & h_2 & u_3 & h_3 & & & & \\ & & & \ddots & \ddots & \ddots & & & \\ & & & & h_{n-3} & u_{n-2} & h_{n-2} & & \\ & & & & & h_{n-2} & u_{n-1} & h_{n-1} & \\ & & & & & & & 1 & \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} 0 \\ v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \\ 0 \end{bmatrix}$$

$$u_i = 2(h_i + h_{i-1})$$

$$v_i = 6(b_i - b_{i-1})$$



Natural Cubic Spline: Conclusion

- 1 Calculate $h_i = t_{i+1} - t_i$, $b_i = \frac{1}{h_i}(y_{i+1} - y_i)$, $u_i = 2(h_i + h_{i-1})$, and $v_i = 6(b_i - b_{i-1})$
- 2 Set up the tridiagonal system

$$\begin{bmatrix} 1 & & & & & \\ h_0 & u_1 & h_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & h_{n-2} & u_{n-1} & h_{n-1} & \\ & & & 1 & & \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} 0 \\ v_1 \\ \vdots \\ v_{n-1} \\ 0 \end{bmatrix}$$

- 3 Solve the system
- 4 Construct the spline function

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + C_i(x - t_i) + D_i(t_{i+1} - x)$$

where

$$D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i \quad C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$$



Algorithm: page 269 NMC7 (page 391, NMC6)

In practice, you do not actually construct a tridiagonal system and solve it (though it is more convenient). It is more efficient to directly solve for z_i in a loop.

See algorithm in the book for efficient implementation.

- 1 Compute for $i = 0, \dots, n - 1$

$$h_i = t_{i+1} - t_i \quad b_i = \frac{1}{h_i}(y_{i+1} - y_i)$$

- 2 Set u, v :
- 3 Tridiagonal solve to get z
- 4 Substitute into the nested form for $S(x)$ equation 12, page 270 NMC7 (NMC6: equation 12 page 392)



Main Point

- Suppose you are given $n + 1$ data points (t_i, y_i)
- Then constructing the natural cubic spline amounts to solving

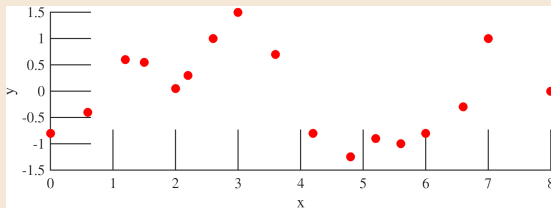
$$\begin{bmatrix} 1 & & & & & & & & \\ h_0 & u_1 & h_1 & & & & & & \\ & h_1 & u_2 & h_2 & & & & & \\ & & h_2 & u_3 & h_3 & & & & \\ & & & \ddots & \ddots & \ddots & & & \\ & & & & h_{n-3} & u_{n-2} & h_{n-2} & & \\ & & & & h_{n-2} & u_{n-1} & h_{n-1} & & \\ & & & & & & 1 & & \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} 0 \\ v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \\ 0 \end{bmatrix}$$

Key Idea

This is a tridiagonal system. Finding the natural cubic spline only costs $\mathcal{O}(n)$ FLOPs.

Runge's Phenomenon

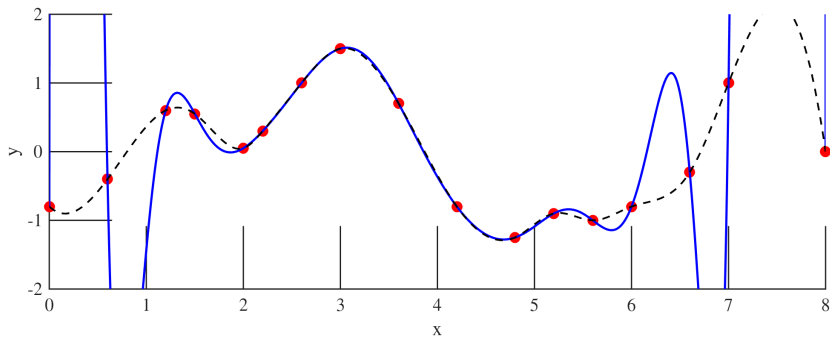
Example - Try interpolation on this data set



- Recall this example
- There are 17 data points
- That means we must fit a degree *sixteen* polynomial



Runge's Phenomenon



- Polynomial interpolation plotted in blue has large oscillations near the end
- Cubic spline interpolation plotted in dashed lines is well-behaved

Key Idea

Cubic splines avoid Runge's phenomenon by using low-degree polynomials locally instead of one high-degree polynomial globally.

3D Data



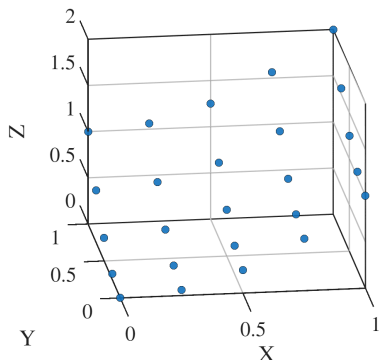
3D Data Points

- 3D data consists of a collection of coordinates

$$(x_i, y_i, z_i).$$

- Example: 25 discrete points on $f(x, y) = x^2 + y^2$

```
1 scatter3(x,y,z)
```

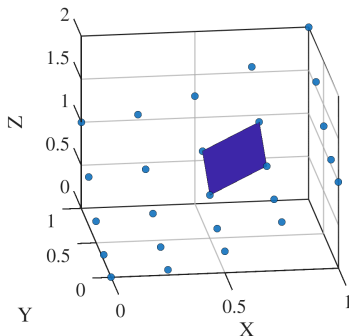


$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$



Face

- A panel created from connecting 4 points is called a **face**



- Plotting all faces creates a surface

Question

How can you tell which 4 points form a face?

Organizing Surface Data

- This format is not great because we cannot determine faces

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$

- This format is better because faces are implicit

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix}, \quad Y = \begin{bmatrix} y_{11} & \cdots & y_{1n} \\ \vdots & \ddots & \vdots \\ y_{m1} & \cdots & y_{mn} \end{bmatrix}, \quad Z = \begin{bmatrix} z_{11} & \cdots & z_{1n} \\ \vdots & \ddots & \vdots \\ z_{m1} & \cdots & z_{mn} \end{bmatrix}.$$

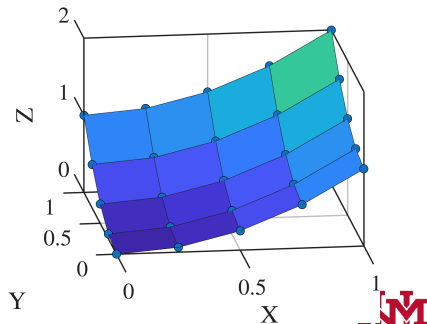
- Use meshgrid in matlab

```
1 x = linspace(0,1,N);  
2 y = linspace(0,1,N);  
3 [X,Y] = meshgrid(x,y);  
4 Z = X.^2 + Y.^2;
```

Visualization

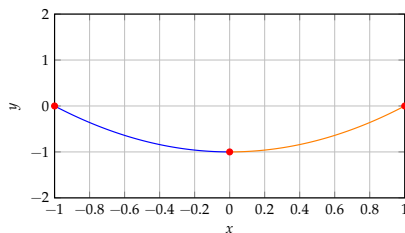
```
1 x = linspace(0,1,N);  
2 y = linspace(0,1,N);  
3 [X,Y] = meshgrid(x,y); % X and Y are matrices  
4 Z = X.^2 + Y.^2;       % Z is also a matrix  
5 surf(X,Y,Z)
```

- Suppose X , Y , and Z are matrices housing rectangular grid of coordinates
- Function `surf(X,Y,Z)` plots all faces based on adjacency

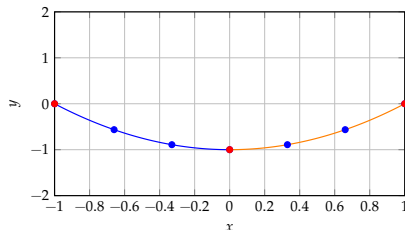


Function `interp1()`

- You may fit a spline on 3 original data points stored in `x` and `y`



- Then `yq = interp1(x,y,xq)` generates extra points (xq_i, yq_i)

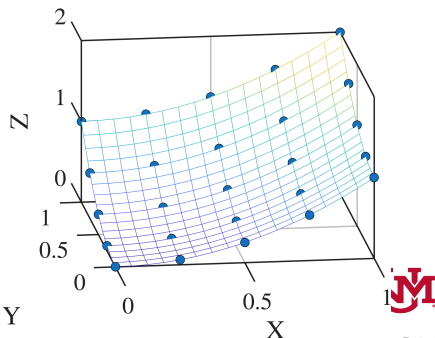
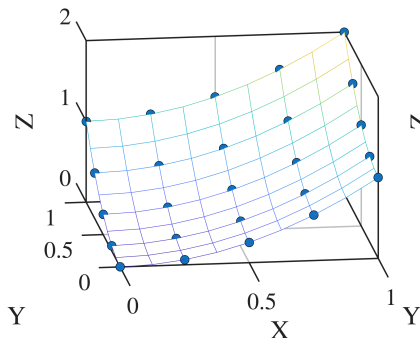


Function interp2()

- The function interp2 works the same way with matrices

```
1 level = 1;  
2 Xq = interp2(X,level,'spline');  
3 Yq = interp2(Y,level,'spline');  
4 Zq = interp2(Z,level,'spline');  
5 mesh(Xq,Yq,Zq)
```

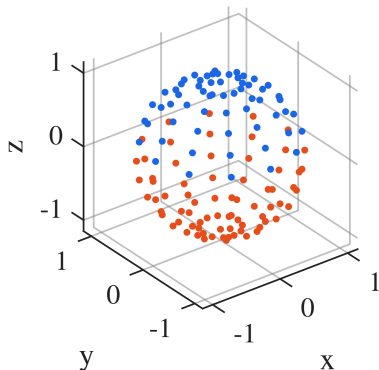
- Parameter level1 controls the refinement



Building an Object from Patches

- A general object can contain multiple surfaces
- Each half is called a **patch**
- Access patch with last index

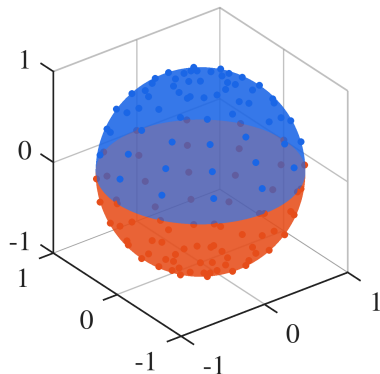
$X(:,:,k)$, $Y(:,:,k)$, $Z(:,:,k)$



```
1 X(:,:,1), Y(:,:,1), Z(:,:,1) % top half of sphere  
2 X(:,:,2), Y(:,:,2), Z(:,:,2) % bottom half
```

Interpolated Sphere

- Interpolate each patch with `interp2`
- Plot the surface of each interpolation with `surf`



Finishing Touches

```
1 shading interp
2 colormap copper
3 axis equal
4 axis off
5 view(135,25)
6 camlight headlight
7 lighting gouraud
```

