Lecture 6: Cubic Splines

Today:

- ***Cubic splines*** for (local) polynomial interpolation

Reminder from last week

Goal: find a function, $f_a(x)$, that approximates a given function, $f(x)$, *accurately* on $x \in [a, b]$

Last time:

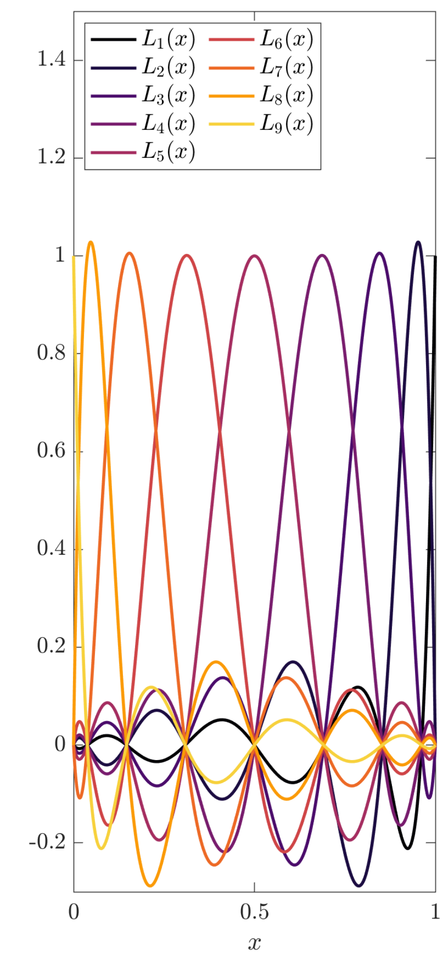
(A) Write $f_a(x)$ as a linear combo of polynomial ***basis*** functions defined *globally* on $x \in [a, b]$:

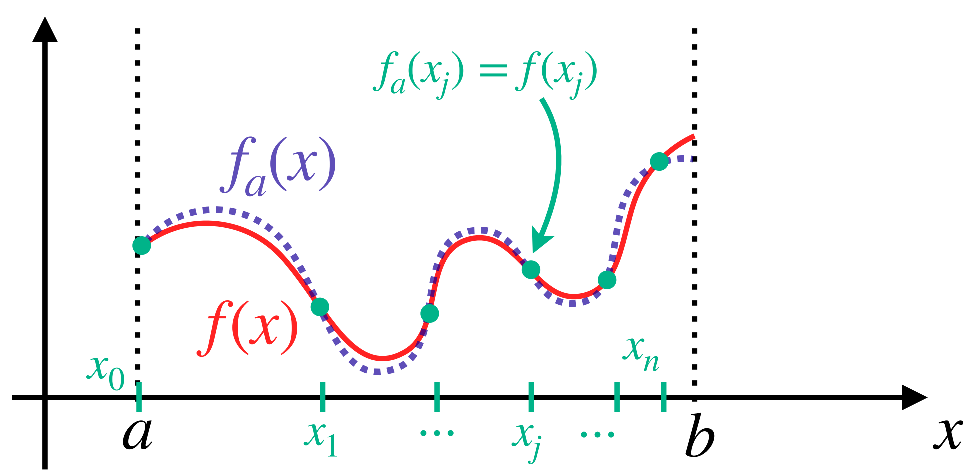
$$f_a(x) = \sum_{j=0}^{n} c_j b_j(x)$$

(B) Solve for the c_j using ***interpolation***

For example

Lagrange basis functions





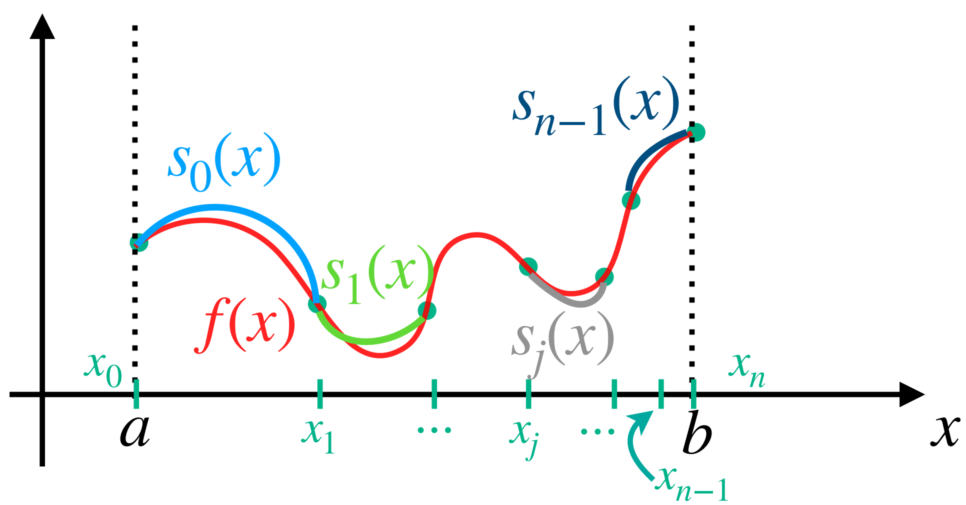
This time:

(A) Key goal: construct $f_a(x)$ by stitching together *locally* defined polynomials

(B) Still use ***interpolation*** to identify coefficients

Local interpolation with splines

Key idea: define $f_a(x)$ in terms of several polynomials defined over a *subset* of $[a, b]$



$$f_a(x) = \begin{cases} s_0(x) & x_0 \leq x \leq x_1 \\ s_1(x) & x_1 \leq x \leq x_2 \\ & \vdots \\ s_j(x) & x_j \leq x \leq x_{j+1} \\ & \vdots \\ s_{n-1}(x) & x_{n-1} \leq x \leq x_n \end{cases}$$

So if we can construct each $s_j(x)$ then we have our approximation $f_a(x)$!

How do we construct each $s_j(x)$? Let's take a concrete example and start with $s_0(x)$…

Activity: what conditions should $s_1(x)$ satisfy?

$$s_0(x_0) = f(x_0)$$
$$s_0(x_1) = f(x_1)$$

Interpolate $f(x)$

$$s_0'(x_1) - s_1'(x_1) = 0$$
$$s_0''(x_1) - s_1''(x_1) = 0$$

Be smooth at transition from s_0 to s_1

Could impose more or less smoothness conditions.
This approach is most common, and leads to ***cubic splines***

Breakout room activity:
What conditions should $s_1(x)$ satisfy?

$s_1(x_1) = f(x_1)$
$s_1(x_2) = f(x_2)$

Interpolate $f(x)$

$s_1'(x_2) - s_2'(x_2) = 0$
$s_1''(x_2) - s_2''(x_2) = 0$

Be smooth at transition from s_1 to s_2

$s_0'(x_1) - s_1'(x_1) = 0$
$s_0''(x_1) - s_1''(x_1) = 0$

Be smooth at transition from s_0 to s_1

Already accounted for by the s_0 conditions we wrote previously. Don't double count!

Writing the general conditions for cubic splines

One can extend the conditions we developed for $s_0(x), s_1(x)$ for the general $s_i(x)$:

(1) $s_i(x_i) = f(x_i), \quad i = 0,\ldots,n-1$

(2) $s_i(x_{i+1}) = f(x_{i+1}), \quad i = 0,\ldots,n-1$

Interpolate $f(x)$

(3) $s_i'(x_{i+1}) - s_{i+1}'(x_{i+1}) = 0, \quad i = 0,\ldots,n-2$

(4) $s_i''(x_{i+1}) - s_{i+1}''(x_{i+1}) = 0 \quad i = 0,\ldots,n-2$

Be smooth at transition from s_i to s_{i+1}

(The smoothness conditions for s_i, s_{i-1} are already accounted for by the equations for $i-1$!)

Great, so we now have the conditions that the $s_i(x)$ must satisfy.

But how do we turn that into a set of equations to actually compute the $s_i(x)$?

Write each $s_i(x)$ in terms of a set of basis functions with unknown coefficients

Use conditions (1)-(4) to solve for the coefficients

But which basis to use?
Could use a Lagrange basis, but we will take a more intuition-based approach for splines

Writing the $s_i(x)$ in terms of unknown coefficients

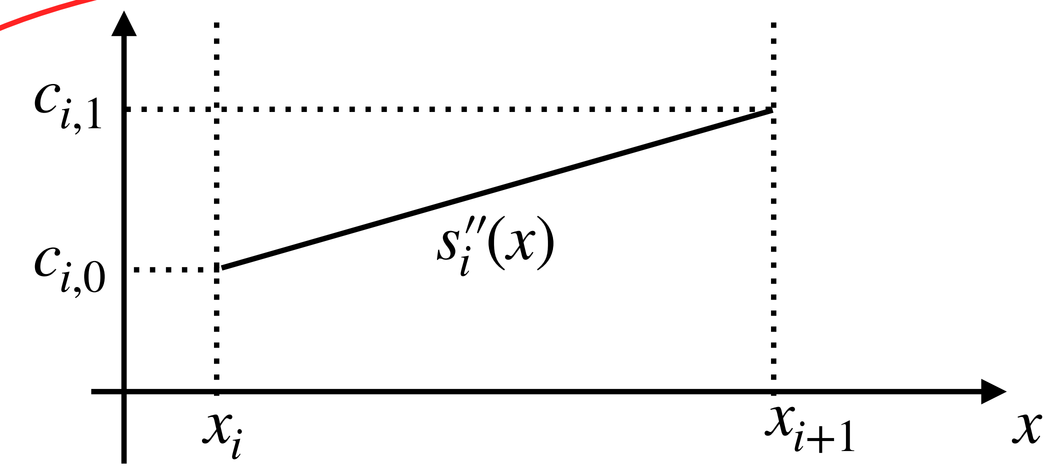
Key observation: $s_i''(x)$ can't vanish after taking two derivatives

$\implies s_i(x)$ must be at least a cubic polynomial

$\implies s_i''(x)$ must be at least a linear polynomial

$$\implies s_i''(x) = \frac{c_{i,0}}{(x_i - x_{i+1})}(x - x_{i+1}) + \frac{c_{i,1}}{(x_{i+1} - x_i)}(x - x_i)$$

A picture helps:



Do not let this notation scare you! This just means "the first coefficient associated with the function s_i"

Some examples:

- For s_1, this coefficient is $c_{1,0}$
- For s_{36}, this coefficient is $c_{36,0}$

We can use this to obtain an expression for $s_i(x)$ by integrating twice:

$$\implies s_i'(x) = \frac{c_{i,0}}{2(x_i - x_{i+1})}(x - x_{i+1})^2 + \frac{c_{i,1}}{2(x_{i+1} - x_i)}(x - x_i)^2 + c_{i,2}$$

$$\implies s_i(x) = \frac{c_{i,0}}{6(x_i - x_{i+1})}(x - x_{i+1})^3 + \frac{c_{i,1}}{6(x_{i+1} - x_i)}(x - x_i)^3 + c_{i,2}x + c_{i,3} \qquad (5)$$

Solving for the unknown coefficients

To solve for the unknown coefficients, we need to set up a linear system with the same number of equations as unknowns

Adding up equations (1)-(4) $\Longrightarrow 4n - 2$ equations

Looking at (5) $\Longrightarrow$ we have $4n$ unknowns

We need 2 more equations! There are several options:

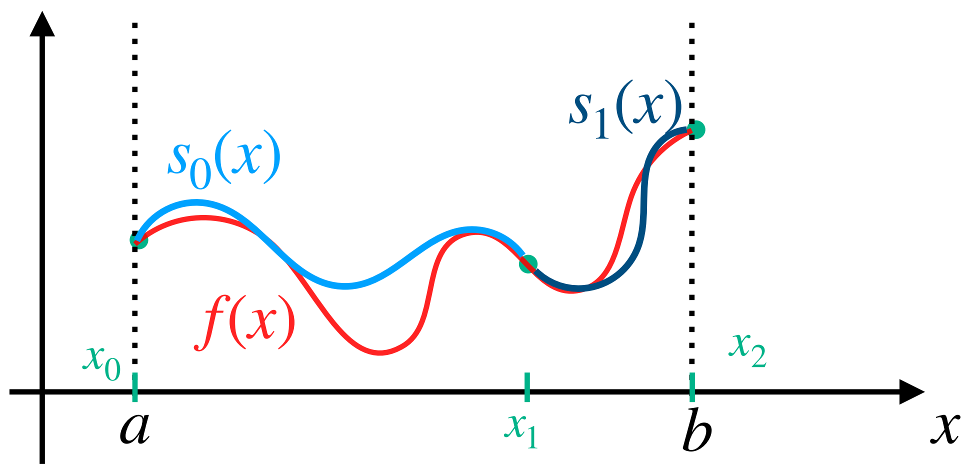
- $s_0'''(x_1) = s_1'''(x_1), \quad s_{n-2}'''(x_{n-1}) = s_{n-1}'''(x_{n-1})$ ← "Not-a-knot"
- $s_0''(x_0) = 0, \quad s_{n-1}''(x_n) = 0$ ← "Natural". Interesting connections to minimizing energy.
- Prescribe some value for $s_0'(x_0)$ and $s_{n-1}'(x_n)$ ← "Complete"

We can use (1)-(4) for each i index along with our choice of 2 extra equations ("Not-a-knot", etc) to come up with a linear system for determining the unknowns!

This can feel a bit abstract. Let's build the linear system for the case of $n = 2$ together…

Cubic splines for $n = 2$

Worked example. For $n = 2$ and assuming *natural splines*, first write out the conditions that must be satisfied in terms of $f(x)$, $s_1(x)$, $s_2(x)$, etc.



$$s_0(x_0) = f(x_0) \qquad (6)$$
$$s_0(x_1) = f(x_1) \qquad (7)$$
$$s_0'(x_1) - s_1'(x_1) = 0 \qquad (8)$$
$$s_0''(x_1) - s_1''(x_1) = 0 \qquad (9)$$
$$s_1(x_1) = f(x_1) \qquad (10)$$
$$s_1(x_2) = f(x_2) \qquad (11)$$
$$s_0''(x_0) = 0 \qquad (12)$$
$$s_1''(x_2) = 0 \qquad (13)$$

Now write out (6) in terms of the unknown coefficients $c_{0,0}$, $c_{0,1}$, etc.

$$s_0(x_0) = \frac{c_{0,0}}{6(x_0 - x_1)}(x_0 - x_1)^3 + \frac{c_{0,1}}{6(x_1 - x_0)}(x_0 - x_0)^3 + c_{0,2}x_0 + c_{0,3} = f(x_0)$$

$$\Longrightarrow \qquad c_{0,0}\frac{(x_0 - x_1)^2}{6} + c_{0,2}x_0 + c_{0,3} = f(x_0) \qquad (14)$$

Can write as Δx^2 if we define $\Delta x = (x_{j+1} - x_j)$

Cubic splines for $n = 2$, cont.

Now start to write the full matrix system and populate the 1st row using eqn (14)

$$\implies \qquad c_{0,0}\frac{(x_0 - x_1)^2}{6} + c_{0,2}x_0 + c_{0,3} = f(x_0)$$

Just re-copying (14) for convenience

$$\implies \begin{bmatrix} \frac{\Delta x^2}{6} & 0 & x_0 & 1 & 0 & 0 & 0 & 0 \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} c_{0,0} \\ c_{0,1} \\ c_{0,2} \\ c_{0,3} \\ c_{1,0} \\ c_{1,1} \\ c_{1,2} \\ c_{1,3} \end{bmatrix} = \begin{bmatrix} f(x_0) \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Activity:

Filling out more of the linear system for $n = 2$

Perform an analogous set of steps as in the previous slides to fill in the 2nd row of the matrix system

The second equation in the linear system we identified was eqn (7), $s_0(x_1) = f(x_1)$

We can write this out in terms of the unknown coefficients $c_{0,0}$, $c_{0,1}$, etc:

$$s_0(x_1) = \frac{c_{0,0}}{6(x_0 - x_1)}(x_1 - x_1)^3 + \frac{c_{0,1}}{6(x_1 - x_0)}(x_1 - x_0)^3 + c_{0,2}x_1 + c_{0,3} = f(x_1)$$

$$\implies \frac{c_{0,1}}{6}\Delta x^2 + c_{0,2}x_1 + c_{0,3} = f(x_1)$$

$$\implies \begin{bmatrix} \frac{\Delta x^2}{6} & 0 & x_0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{\Delta x^2}{6} & x_1 & 1 & 0 & 0 & 0 & 0 \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} c_{0,0} \\ c_{0,1} \\ c_{0,2} \\ c_{0,3} \\ c_{1,0} \\ c_{1,1} \\ c_{1,2} \\ c_{1,3} \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Fun fact. This approach for computing the $c_{i,j}$ coefficients is actually ***not*** the most efficient. There is a variant of the linear system that is faster to solve. It is not necessary to take this faster approach in class, but the interested student is referred to *Chapra & Canale, McGraw Hill, pp. 516-517.*

Zooming out conceptually: where are we at with splines

I hope you now have some intuition for how the linear systems are built to solve for the coefficients $c_{i,j}$, $(i = 0,\ldots n-1;\ j = 0,1,2,3)$

Once we have the coefficients we have our cubic spline interpolant $f_a(x)$!

- Equation (5) lets us write each $s_i(x)$, $i = 0,\ldots,n-1$, once we have these coefficients
- Then we write $f_a(x)$ by "stitching" together the different $s_i(x)$, as shown in slide (3)

Note. We only talked about building the linear system for the coefficients $c_{i,j}$, $(i = 0,\ldots n-1;\ j = 0,1,2,3)$ when $n = 2$.

How do we automate the process for general n? HW 2 will cover this!