6/3/14

- Recall: $p(x_i) = f(x_i) \quad i=0,\dots,n$

  p polynomial of degree n

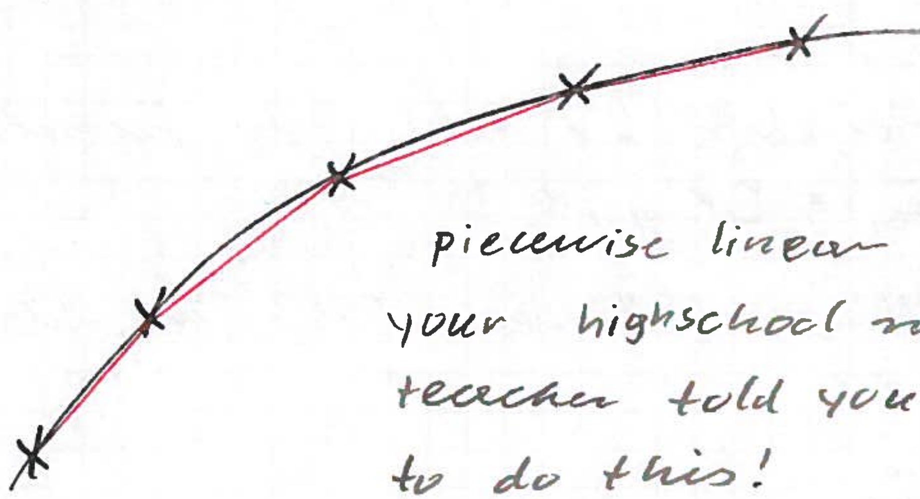
- Then

$$f(x) - p(x) = \frac{1}{(n+1)!} \prod_{i=0}^{n} (x - x_i)\, f^{(n+1)}(\xi) \qquad (*)$$

- The error can increase as the polynomial degree increases.

- In general, interpolation by polynomials of a high degree is a bad idea

- Alternative: piecewise polynomial interpolation



piecewise linear

your highschool math teacher told you not to do this!

- Nonetheless, let's press ahead

Divide the interval $[a,b]$ into subintervals

$$a = x_0 < x_1 < x_2 < \cdots < x_N = b$$

- The nodes, knots, or abcissas, x_i may or may not be evenly spaced.

$$h_i = x_i - x_{i-1} \qquad i = 1, \cdots, N$$

$$h = \max h_i$$

evenly spaced if all $h_i = h$

- Suppose we are also given data $y_i = f(x_i)$ $i = 0, \cdots, n$

- Idea 1. Interpolate by a linear function on each $I_i = [x_{i-1}, x_i]$

- piecewise linear, "broken line," an old idea.

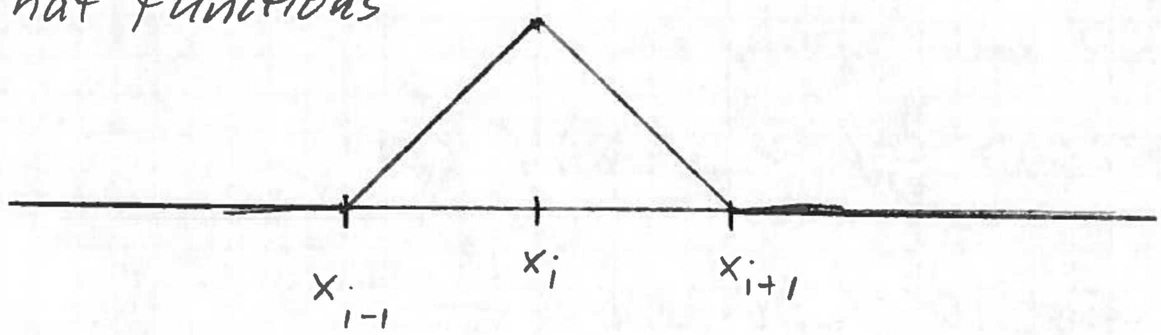
- we can write

$$L(x) = \sum_{i=0}^{n} y_i L_i(x)$$

where L_i is linear in each I_j, $L_i(x_i) = 1$, $L_i(x_j) = 0 \quad i \neq j$

$$L_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i=j \\ 0 & \text{if } i \neq j \end{cases}$$

- The graphs of the L_i are well-known "hat functions"



- The interpolant is in cardinal form
- The data serve as coefficients
- corresponds to the Lagrange form of the interpolating polynomial
- Note that the hat functions have small support. They are non-zero only on two intervals.
- By contrast the Lagrange basis functions for polynomial interpolation have $[a,b]$ as their support.

- Suppose $x \in [x_{i-1}, x_i] = I_i$
- what is the error?
- On I_i we have just a linear interpolant and we can apply (*)

$$f(x) - L(x) = \frac{1}{2!}(x-x_{i-1})(x-x_i) f''(\xi)$$

- Note that $|(x-x_{i-1})(x-x_i)| \le \frac{h^2}{4}$
- suppose that $|f''(x)| \le M_2$ in $[a,b]$
- Then

$$|f(x) - L(x)| \le \frac{h^2}{8} M_2 \quad \text{for all } x \text{ in } [a,b]$$

- The error goes to zero like $O(h^2)$
- Increasing the number of data sites and decreasing h reduces the error.
- That's good
- Not so good: - graph is only continuous
  - $O(h^2)$ is not overly fast

- Idea 2: interpolate the derivative as well.

- This will increase the speed of convergence and make the graph C^1 but of course it will require derivative values

- This gives rise to "piecewise cubic Hermite" interpolants:

Data: $y_i = f(x_i)$ $\quad \bar{y}_i = f'(x_i)$

$H(x)$ = cubic on each $[x_{i-1}, x_i]$

and

$$H(x_i) = y_i \qquad H'(x_i) = \bar{y}_i \qquad i = 0, \dots, n$$

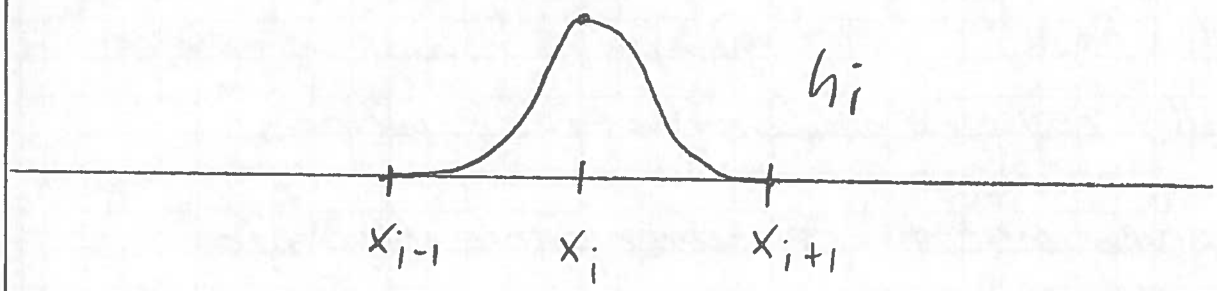
$$H(x) = \sum_{i=0}^{n} \left(y_i h_i(x) + \bar{y}_i \bar{h}_i(x) \right)$$

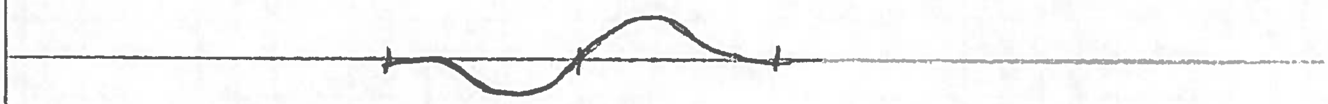
Where the h_i and $\bar{h}_i$ are cubic on each $[x_{i-1}, x_i]$ and

$$h_i(x_j) = \delta_{ij} \qquad h_i'(x_j) = 0$$

$$\bar{h}_i(x_j) = 0 \qquad \bar{h}_i'(x_j) = \delta_{ij}$$

The graphs of the h_i and $\bar{h}_i$ look like this:





- again we have small supports.
- Exercise: Find algebraic expressions for the L_i, h_i, and $\bar{h}_i$
- what's the error on $[x_{i-1}, x_i]$

(*) can be modified. We get

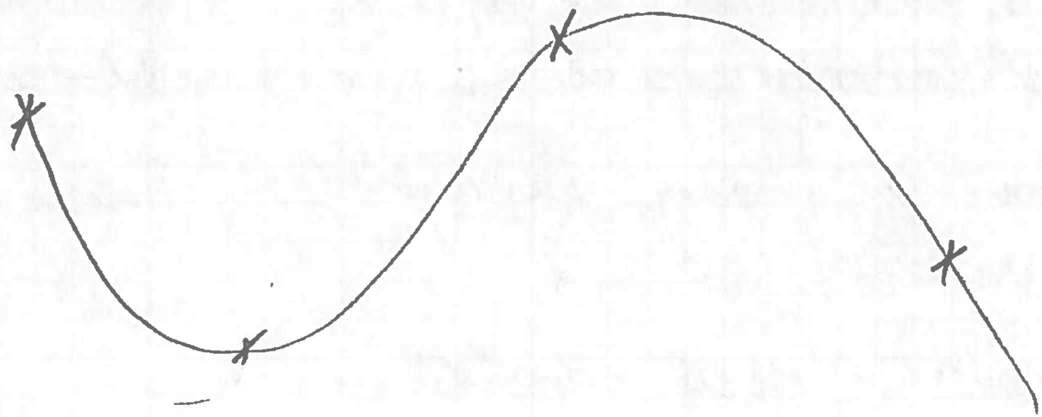
$$f(x) - H(x) = \frac{1}{4!}(x-x_{i-1})^2(x-x_i)^2 f^{(4)}(\xi)$$

Thus

$$|f(x) - H(x)| \leq \frac{h^4}{16 \cdot 24} M_4 = \frac{M_4 h^4}{384}$$

when $M_4 = \max_{x \in [a,b]} |f^{(4)}(x)|$

- But we need derivatives.
- popular alternative
- cubic splines



- motivated by mechanical analogy: fit an elastic wire through given points.
- Approximating this mathematically gives a cubic spline

$$S(x_i) = y_i$$

$$S \text{ in } C^2[a,b]$$

S cubic on each interval $[x_{i-1}, x_i]$

- let's count conditions and parameters

4 N parameters

$s(x_i) = y_i$ $2+2(N-1)=2N$ conditions

s' continuous at $x_1, \dots x_{N-1}$ $N-1$ conds

s'' continuous at $x_1, \dots x_{N-1}$ $N-1$ conds

- have 2 more parameters than conditions

- impose 2 end conditions

forced end: $s'(a) = A$ $s'(b) = B$

natural $s''(a) = s''(b) = 0$

not-a-knot s''' continuous at x_1 and x_{N-1}

- cardinal splines $S_i(x_j) = \delta_{ij}$ have full support, all of $[a,b]$ (except knots)

- Error analysis much more complicated

- Carl de Boor A practical guide to splines, Springer Verlag 1978

- built into Matlab