Hermite Interpolation

Sunday, September 20, 2020 11:24 PM

Brief lecture!

Our setup: $\{x_0, x_1, \dots, x_n\}$ nodes, with values $\{y_0, y_1, \dots, y_n\}$

and often $y_i = f(x_i)$ for some true "underlying" function f.] we'll assume this.

What if instead of just asking for a polynomial p such that $p(x_i) = f(x_i)$ $i = 0, 1, \dots, n$

we also want $p'(x_i) = f'(x_i)$?

(Why do this? Better capture shape of f)

This is known as HERMITE INTERPOLATION

In fact, you could ask to match $p''(x_1) = f''(x_1)$, $p'(x_1) = f'(x_1)$, $p(x_1) = f(x_1)$ and just $p(x_0) = f(x_0)$, $p(x_2) = f(x_2)$

The general case is finding the OSCULATING POLYNOMIAL

SIMPLE RULE-OF-THUMB (which is 100% correct):

The number of coefficients in the polynomial must match the number of constraints

(recall a degree n polynomial has $n+1$ coefficients)

$n=2$, $\underbrace{a_2 x^2 + a_1 x + a_0}_{3 \text{ coefficients}}$

Ex:

1) Standard interpolation on $\underbrace{\{x_0, x_1, \dots, x_n\}}_{n+1}$ pts. so $n+1$ constraints $p(x_i) = f(x_i)$, $i = 0, 1, \dots, n$

... so, degree n polynomial

2) Hermite on $\{x_0, x_1, \dots, x_n\}$, $p(x_i) = f(x_i)$, $p'(x_i) = f'(x_i)$ $i = 0, 1, \dots, n$ so $2(n+1)$ constraints

... so, degree $2n+1$ polynomial

How to find the Hermite interpolating polynomial

Recall our Lagrange interpolating polynomial of degree n:

$$L_{n,k}(x) = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{(x - x_i)}{(x_k - x_i)} \quad \text{for } k = 0, 1, \dots, n$$

and $L_{n,k}(x_i) = \begin{cases} 1 & i = k \\ 0 & \text{else} \end{cases}$

Thm 3.9

$f \in C^1([a,b])$ and $\{x_0, x_1, \dots, x_n\} \subseteq [a,b]$ are distinct, then the unique polynomial of least degree that matches f and f' on $\{x_0, \dots, x_n\}$ is the Hermite polynomial of degree at most $2n+1$ given by

① $$H_{2n+1}(x) := \sum_{j=0}^{n} \left(f(x_j) H_{n,j}(x) + f'(x_j) \hat{H}_{n,j}(x) \right)$$

where $H_{n,j}(x) = \left(1 - 2(\underbrace{x - x_j}_{\text{degree } 1}) L'_{n,j}(x_j)\right) \cdot \underbrace{L^2_{n,j}(x)}_{\text{degree } 2n}$ ... So degree $\leq 2n+1$

and $\hat{H}_{n,j}(x) = (x - x_j) L^2_{n,j}(x)$.

② and if $f \in C^{2n+2}([a,b])$ then $\forall x \in [a,b]$, $\exists \xi \in (a,b)$ such that

$$f(x) = H_{2n+1}(x) + \frac{(x-x_0)^2(x-x_1)^2 \cdots (x-x_n)^2}{(2n+2)!} f^{(2n+2)}(\xi)$$

---

proof sketch

① Note if $i \neq j$ then $H_{n,j}(x_i) = \hat{H}_{n,j}(x_i) = 0$ thanks to the $L_{n,j}(x)$ terms

and if $i = j$ then $H_{n,j}(x_j) = 1$, $\hat{H}_{n,j}(x_j) = 0$

So $H_{2n+1}(x_i) = f(x_i)$ holds.

Can similarly show
$$\left.\begin{aligned} &H'_{n,j}(x_i) = \hat{H}'_{n,j}(x_i) = 0 \\ &H'_{n,j}(x_j) = 0 \quad \hat{H}'_{n,j}(x_j) = 1 \quad \text{] swap} \end{aligned}\right\} \text{same but for derivatives.}$$
via explicit calculation.

To show uniqueness, let p and q be two such polynomials, then consider the roots of $d = p - q$ and d'

② Similar to Lagrange error term (generalized Rolle's thm, like MVT)

Efficient ways to compute

There is a divided difference formula, but don't bother learning it (too specialized). We'll mainly use for splines where we keep n small.