Week 9: Wednesday, Mar 28

Summary of last time

We spent most of the last lecture discussing three forms of polynomial interpolation. In each case, we were given function values $\{y_i\}_{i=0}^d$ at points $\{x_i\}_{i-1}^d$, and we wanted to construct a degree d polynomial such that $p(x_i) = y_i$. We do this in general by writing

$$p(x) = \sum_{j=0}^{d} c_j \phi_j(x),$$

where the functions $\phi_j(x)$ form a basis for the space of polynomials of degree at most d. Then we use the interpolation conditions to determine the coefficients c_j via a linear system

$$Ac = y,$$

where $A_{ij} = \phi_j(x_i)$. In the last lecture, we considered three choices of basis functions $\phi_j(x)$:

1. Power basis:
$$\phi_j(x) = x^j.$$
2. Lagrange basis:
$$\phi_j(x) = \frac{\prod_{i \neq j}(x - x_i)}{\prod_{i \neq j}(x_j - x_i)}.$$
3. Newton basis:
$$\phi_j(x) = \prod_{i<j}(x - x_i).$$

The power basis yields an ill-conditioned system matrix (the Vandermonde matrix). The Lagrange basis leads to a trivial linear system, but it takes $O(d)$ time to evaluate each Lagrange polynomial and so $O(d^2)$ time to evluate the interpolant. The Newton basis is a nice compromise: the coefficients can be computed in $O(d^2)$ time as the solution to an upper triangular system or through a divided difference recurrence, and the polynomial itself can be evaluated in $O(d)$ time using an algorithm like Horner's rule.

Divided differences and derivatives

The coefficients in the Newton form of the interpolant are *divided differences*. For a given function f known at sample points $\{x_i\}_{i=1}^n$, we can evaluate divided differences recursively:

$$\begin{aligned} f[x_i] &= f(x_i), \\ f[x_i, x_{i+1}, \ldots, x_j] &= \frac{f[x_i, x_{i+1}, \ldots, x_{j-1}] - f[x_{i+1}, \ldots, x_j]}{x_i - x_j}. \end{aligned}$$

This recurrence is numerically preferable to finding the coefficients of the Newton interpolant by back substitution.

You might recognize the first divided difference $f[x_1, x_2]$ as a derivative approximation. In fact, if f is a differentiable function, then the mean value theorem tells us that $f[x_1, x_2] = f'(\xi)$ for some ξ between x_1 and x_2. Thus if f is a continuously differentiable function, it makes sense to define

$$f[x_i, x_i] \equiv f'(x_i).$$

This gives us a natural way to solve *Hermite* interpolation problems in which we specify both function values and derivatives at specified points.

More generally, it turns out that if $f \in C^{m-1}$, then

$$f[x_1, x_2, \ldots, x_m] = \frac{f^{(m-1)}(\xi)}{(m-1)!}, \quad \text{some } \xi \in (\min\{x_i\}, \max\{x_i\})$$

Therefore, in the limiting case as we let all the x_j approach some common point x_0, the Newton form of the interpolant degenerates into a Taylor approximation.

Error in polynomial approximation

The relation between divided differences and derivatives is incredibly useful in reasoning about how well polynomial interpolants approximate an underlying function. Suppose we approximate $f \in C^n$ by a polynomial p of degree $n-1$ that interpolates f at points $\{x_i\}_{i=1}^n$. At any point x, we can write $f(x) = p^*(x)$, where $p^*(x)$ is the degree n polynomial interpolating f at

$\{x_i\}_{i=1}^n \cup \{x\}$. This may seem somewhat silly, but it gives us the error representation

$$\begin{aligned} f(x) - p(x) &= p^*(x) - p(x) \\ &= f[x_1, \ldots, x_n, x] \prod_{i=1}^n (x - x_i) \\ &= \frac{f^{(n)}(\theta)}{n!} \prod_{i=1}^n (x - x_i). \end{aligned}$$

If x lies within h of all the values x_i and $|f^{(n)}| \leq M_n$ on the interval bounded by the points in question, then we have

$$|f(x) - p(x)| \leq \frac{M_n h^n}{n!}.$$

This bound suggests that high-order polynomial interpolation of a smooth function over a bounded interval can provide very accurate approximations to the function values, with two catches. First, the h^n term may not be small (especially in *extrapolation*, where x lies outside the convex hull of the data points). Second, M_n may grow quickly as a function of n. Note that these two effects are not independent; for example, we can scale the nodal coordinates to make h smaller, but then M_n gets commensurately bigger. The standard example of these effects, due to Runge, is the function

$$\phi(t) = \frac{1}{1 + 25t^2}.$$

Polynomial approximations to $\phi(t)$ by interpolation on a uniform mesh on $[-1, 1]$ oscillate wildly toward the end points of the interval, and it is not true in this case that ever higher-degree interpolating polynomials provide ever-better function approximations. This is a general problem, known as the *Runge phenomena*, and there are two standard fixes. The first fix is to use something other than polynomials (piecewise polynomial functions are particularly popular). We will talk about this option next week. The second approach involves optimizing the location of the sample points, a topic which we will turn to now.

Chebyshev interpolation

Suppose we want a polynomial interpolant that accurately represents some function on a bounded interval. Earlier, we showed that

$$f(x) - p(x) = \frac{f^{(n)}(\theta)}{n!} \prod_{i=1}^{n} (x - x_i).$$

If $|f^{(n)}(x)| \leq M$ on the interval $[a, b]$, then

$$|f(x) - p(x)| \leq \frac{M}{n!} \prod_{i=1}^{n} (x - x_i).$$

So one natural approach to trying to build accurate interpolants is to try to minimize some norm that measures the size of

$$\psi(x) = \prod_{i=1}^{n} (x - x_i)$$

over the interval $[a, b]$. If we care about the values pointwise, it makes sense to try to choose the interpolation points to minimize

$$\|\psi\|_{L^\infty([a,b])} = \max_{x \in [a,b]} |\psi(x)|.$$

This leads to the choice of *Chebyshev points* on $[-1, 1]$

$$\xi_i = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i = 1, \ldots, n.$$

For more general intervals, we can simply apply an affine mapping to get the interpolation points

$$x_i = a + \frac{b-a}{2}(\xi_i + 1).$$

If we choose the Chebyshev points as interpolation nodes, then

$$\|\psi\|_{L^\infty([a,b])} = 2^{1-n}$$

and so we have the error bound

$$|f(x) - p(x)| \leq \frac{M}{2^{n-1} n!}$$

for $x \in [a, b]$.

Problems to ponder

1. Suppose $p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$. Write a linear system for the coefficients a_j such that $p(0) = p_0$, $p'(0) = q_0$, $p(1) = p_1$, $p'(1) = q_1$.

2. For the points $x_1 = -1, x_2 = 0, x_3 = 1$ and the values of $y_1 = 0, y_2 = 1, y_3 = 1$, write the interpolating polynomial in power form, Lagrange form, and Newton form.

3. In lecture, I described Horner's rule for evaluating a polynomial
$$p(x) = \sum_{j=0}^{d} c_j x^j$$
in terms of the recurrence
$$\begin{aligned} p_{d+1}(x) &= 0 \\ p_j(x) &= x p_{j+1}(x) + c_j. \end{aligned}$$
What is the equivalent recurrence for evaluating the Newton form of the interpolant?

4. Describe how to find coefficients c_i such that
$$g(x) = c_1 + c_2 x + c_3 \sin(x) + c_4 \cos(x)$$
interpolates $f(x)$ at distinct points x_1, x_2, x_3, x_4.

5. In class, we wrote the Lagrange interpolant as
$$p(x) = \sum_{j=1^n} y_i L_i(x)$$
where $L_i(x) = \prod_{k \neq i}(x - x_k)$. The book describes computation of the Lagrange interpolant via the barycentric formula
$$p(x) = \frac{\sum_{j=1^n} w_j y_j/(x - x_j)}{\sum_{j=1^n} w_j/(x - x_j)}$$
where $w_j^{-1} = \prod_{i \neq j}(x_j - x_i)$. Why are these formulas equivalent, and what is the advantage of barycentric interpolation?