Math 128A, Summer 2019
Lecture 10, Tuesday 7/9/2019

Topics Today:
- Deriving Interpolation Error
- Runge Phenomenon
- Chebyshev Points (as a Polynomial)

1 Review: Error in Polynomial Interpolation

Recall that for degree n, we take $(n+1)$ distinct points at which to interpolate (match values) between $p(x)$ and $f(x)$. We want to essentially predict the behavior (or values) of f by knowing other values of $p(x_i) = f(x_i)$.

$$\begin{aligned} p(t) &= a_0 + a_1 t + a_2 t^2 + \cdots + a_n t^n \\ &= b_0 + b_1(t - t_0) + b_2(t - t_0)(t - t_1) + \cdots + b_n(t - t_0)\cdots(t - t_{n-1}), \end{aligned}$$

where this second (Newton) basis gives some triangular system of equations. This is good because we can look at them and immediately know if we have a unique solution. Or, we can write it via Lagrange as:

$$p(t) = f_0 L_0(t) + \cdots + f_n L_n(t),$$

and for **square** linear systems $(n+1) \times (n+1)$, if we have a solution, we are guaranteed (for free) that it is unique.
(That is, for a given function and a set of points at which to interpolate, regardless our interpolation basis or algorithm, our interpolating polynomial is unique. Hence we deduce the error from interpolation should be unique.)
Recall:

$$L_j(t) = \prod_{k \neq j} \left[\frac{t - t_k}{t_j - t_k}\right]$$

$$L_j(t_k) = \delta_j, k = \begin{cases} 1, & j = k \\ 0, & j \neq k \end{cases}$$

And hence:

$$\begin{aligned} L_j(t) &= \lambda_j \omega(t) \frac{1}{t - t_j} \\ \lambda_j &= \Pi_{k \neq j} \frac{1}{t_j - t_k} \\ \omega(t) &= (t - t_0)(t - t_1)\cdots(t - t_n) \end{aligned}$$

And so $\omega(t)$ is a polynomial with zeros precisely at $t_0, t_1, \ldots, t_n$. This polynomial tells us if our distribution (spread) of distinct points is good for our purposes. We want to make the values of $|\omega(t)|$ small-valued for the intervals from which we are estimating.
Our error is zero at each of the interpolation points, and our $\omega(t)$ is also zero at each of the interpolation points. Hence:

Polynomial Interpolation Error:

$$f(t) - p(t) =$$
$$\left(\underbrace{\frac{1}{(n+1)!}}_{\text{normalizes when } f(t)=\omega(t)}\right) \cdot \left(\underbrace{f^{(n+1)}(\xi)}_{\text{0, if } f \text{ deg } n \text{ poly}}\right) \cdot \left(\underbrace{\omega(t)}_{\text{0 at interpolating points}}\right).$$

The middle part is just to make sure that interpolating an n-degree f using $n+1$ polynomial p would cause the error to be exactly zero, because the polynomial will be unique (and exactly equal to f everywhere).
Remember that when taking the derivatives of $\omega(t)$, there's chain rule, but only one term counts.
We can prove this error formula, but for now we just 'construct' a polynomial that works for the interpolation error.
So, for linear interpolation, we use the above:

$$f(t) - \left[f(t_0) + \frac{f(t_1) - f(t_0)}{t_1 - t_0}(t - t_0)\right] = \frac{1}{2!} f''(\xi)(t - t_0)(t - t_1)$$

Example: $f(t) := e^t$ for $t = 0, 1, 2$. Then $f(0) = 1, f(1) = e, f(2) = e^2$. We'll use the Lagrange basis polynomial:

$$p(t) := \frac{(t-1)(t-2)}{(0-1)(0-2)} \cdot 1 + \frac{(t-0)(t-2)}{(1-0)(1-2)} \cdot e + \frac{(t-0)(t-1)}{(2-0)(2-1)} e^2$$

We can multiply this our and give it in the monomial basis, but this is more than sufficient. Strain says to just write out the Lagrange form, and you're good to go.
Then the error will be:

$$f(t) - p(t) = \frac{1}{3!} \cdot f^{(3)}(\xi) \cdot [(t-0)(t-1)(t-2)]$$

And we say: $e^\xi \in \left[e^0, e^2\right]$. So a worst-case a bound for our error can be:

$$|f(t) - p(t)| \leq \frac{1}{6} e^2 \max\{(t)(t-1)(t-2)\}$$

To find the maximum, consider:

$$g(t) = t(t-1)(t-2)$$
$$g'(t) = 0$$

So we should look at how the choice of interpolation points affects the error bounds $|\omega(t)|$.
We want to find the smallest $\max|\omega(t)| = |(t - t_0) \cdots (t - t_n)|$. To make it simple, just take some finite interval:

$$a \leq t_j \leq b.$$

We take $n+1$ intervals and place them on this interval $[a, b]$. And of course, shrinking the interval between the interpolation points makes the maximum

smaller. A bad way to distribute points is equispaced (but sometimes this is all we can do, because the data is given at equispaced points, like stock market data at every interval of time). So what is the best we can do, if we are able to decide how to distribute the points?
Consider: $t_0 = 0, t_1 = 1, \ldots, t_n = n$. Then for $0 \leq t \leq \frac{1}{2}$,

$$\omega(t) = (t-0)(t-1)(t-2)\cdots(t-n) \approx n!.$$

And similarly, for $\frac{n}{2} \leq t \leq \frac{n+1}{2}$,

$$\omega(t) \approx \left(\left(\frac{n}{2}\right)!\right)^2.$$

To view this, consider from Stirling's:

$$\begin{aligned} n! &\approx \left(\frac{n}{e}\right)^n \\ \left[\left(\frac{n}{2}\right)!\right]^2 &\approx \left[\left(\frac{n}{2e}\right)^{n/2}\right]^2 = \left(\frac{n}{2e}\right)^n \omega(t) && \approx 2^{-n} \cdot n! \end{aligned}$$

So for $n = 20$, the error bound is a million (2^{20}) times larger near the end points than in the center of the interval. We call this the "Runge Phenomenon", (pronounced 'roon-huh').

Runge Phenomenon - Mantra:
High-degree polynomial interpolation at equispaced points is **wildly inaccurate** near the ends of the (interpolating) interval.

Consider that no one is making us do high-degree interpolation, so we can do piecewise interpolation with lower degree polynomials. And perhaps instead of polynomials, we may use our favorite functions (like fourier or splines). Or, instead of exact interpolations, we can just keep a least-squares fit near (but not exact) to the polynomial. We can also use Chebyshev points to get around equispaced points. Or, we can just interpolate on a larger interval but only pull (test) on the middle of the region, where our error is low.

2 Chebyshev Points

These are very simple and completely eliminate the Runge Phenomenon. The central idea is to build $\omega(t)$ which has $n+1$ zeroes on $[-1, 1]$ (convention), where we make all the maxima and minima the same height, $\pm$. We call this "equioscillating".
We simply construct polynomials T that guarantee our desired condition that $|\sup T| = |\inf T|$. That is, we want to bound the polynomial function min and max values to precisely hit $[-1, 1]$.

$$\begin{aligned} T_0(t) &:= 1 \qquad \text{(interpolating (1,1))} \\ T_1(t) &:= t \qquad \text{(interpolating (-1,-1) and (1,1))} \\ &\qquad \text{(now interpolating (-1,1), (0,-1), (1,1)):} \\ T_2(t) &:= 1\frac{(t-0)(t-1)}{(-1-0)(-1-1)} + (-1)\frac{(t-1)(t-(-1))}{(0-1)(0-(-1))} + 1\frac{(t-(-1))(t-0)}{(1-(-1))(1-0)} \\ &= 2t^2 - 1 = 2t \cdot T_1(t) - T_0(t) \end{aligned}$$

But equivalently, defining $\cos x := t; x := \cos^{-1}(t)$, this is:

$$T_2(t) := \cos(2t) = 2\cos^2(x) - 1,$$

and we easily check our definition works for T_1, T_0 as above. Generalizing, we have as a candidate for an equioscillating polynomial:

$$T_k(t) = \cos(kx).$$

So we say: $|t| \leq 1 \iff 0 \leq x \leq \pi$. Now the only 'scary' part is that we don't know this is a polynomial (it certainly doesn't look like it).
Recall:

$$\begin{aligned}
\cos(a+b) &= \cos a \cos b - \sin a \sin b \\
\cos(a-b) &= \cos a \cos b + \sin a \sin b \\
\cos(a+b) + \cos(a-b) &= 2\cos b \cos a \quad \text{(adding the above equations)}
\end{aligned}$$

And letting $a := bx, b := x$, we get the following recurrence relation:

$$T_{k+1}(t) + T_{k-1} = 2t\ T_k(t).$$

Or in a more useful form:

$$T_{k+1} = 2t\ T_k - T_{k-1}$$

And so,

$$\begin{aligned}
T_0 &= 1, \qquad T_1 = t, \\
T_2 &= 2t(t) - 1 \\
T_3 &= 2t\ (2t^2 - 1) - t \\
&= 4t^3 - 3t \implies t = \left\{0, \frac{\sqrt{3}}{2} \frac{-\sqrt{3}}{2}\right\}
\end{aligned}$$

And we can keep going onwards to generate polynomials.
A conjecture can be that T_k is even iff k is even, and similarly T_k is odd if k is odd. Consider the following definition:

$$\begin{aligned}
T_k(t) &= \cos(kx) = 0 \\
kx &= (2j+1)\pi/2, \qquad j = 0, 1, \ldots, (k-1) \\
\implies x &= \frac{(2j+1)\pi}{2k}
\end{aligned}$$

where this is a polynomial in disguise. And we write:

$$t_j = \cos\left(\frac{2j-1}{2k}\pi\right); \qquad 1 \leq j \leq k$$

so these guys are numbered backwards, from right to left, which is ok (because we could add interpolation points in any order).
So effectively, we took a circle, divide the angle of 0 to π evenly, and project the points on the circle down onto the x-axis.

Remark: Using Chebyshev points is a good way to distribute points when we don't know the function we are interpolating.

From the definition of our recurrence relation

$$T_{k+1} := 2t\ T_k - T_{k-1}; \qquad T_0 := 1, T_1 := t$$

we can see that T_k must be a k-degree polynomial. We look for the characteristic equation (which happens because we look for T_k as a linear combination of two powers r^k, the roots of the characteristic equation). Then:

$$\begin{aligned} T_k &= r^k \\ r^{k+1} &= (2t)r^k - r^{k-1} \quad \text{(from the recurrence relation)} \\ r^2 + 2tr + 1 = 0 &\quad \text{(dividing by } r^{k-1}) \\ \implies r &= t \pm \sqrt{t^2-1} = t \pm i\sqrt{1-t^2}. \end{aligned}$$

Because we have two roots, we then have:

$$T_k = ar_+^k + br_-^k$$

And we set:

$$\begin{aligned} T_0 &= 1 = a + b \\ T_1 &= ar_+ + br_- = t \end{aligned}$$

Solving this our favorite way (i.e. 2×2 matrix), we have:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{-2\sqrt{t^2-1}} \begin{bmatrix} r_- - t \\ -r_+ + t \end{bmatrix}$$

where:

$$\begin{aligned} r_- &= t - \sqrt{t^2-1} \quad &\implies \quad r_- - t &= -\sqrt{t^2-1} \\ r_+ &= t + \sqrt{t^2-1} \quad &\implies \quad r_+ - t &= -\sqrt{t^2-1} \end{aligned}$$

And we conclude:

$$\begin{aligned} T_k(t) &= \frac{1}{2}\left[\left(t + \sqrt{t^2-1}\right)^k + \left(t - \sqrt{t^2-1}\right)^k\right] \\ &= \frac{1}{2}\left[e^{ikx} + e^{-ikx}\right] = \cos(kx), \end{aligned}$$

and we have that cosine is precisely a polynomial satisfying our requirements for T_k (recall we set $t := \cos(kx)$).

Lecture ends here.