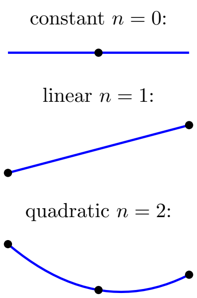
Math 4073: Polynomial Lagrange Interpolation

Interpolation is the filling-in of missing data; from just a few samples of an otherwise unknown function we try to reconstruct that unknown function. In some sense this must be impossible but nevertheless we can do very well in practice! We start by looking at the Lagrange interpolating polynomial.[1]



Setup and notation Let $\Pi_n = \{$real polynomials of degree $\leq n\}$. Problem: given scalar data f_i at distinct scalar x_i, $i = 0, 1, \ldots, n$, with $x_0 < x_1 < \cdots < x_n$, can we find a polynomial p_n such that $p_n(x_i) = f_i$? Such a polynomial is said to *interpolate* the data.

Theorem: $\exists p_n \in \Pi_n$ such that $p_n(x_i) = f_i$ for $i = 0, 1, \ldots, n$.

Proof (Constructive!) Consider, for $k = 0, 1, \ldots, n$, the "cardinal polynomial"

$$L_{n,k}(x) = \frac{(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)} \in \Pi_n. \tag{1}$$

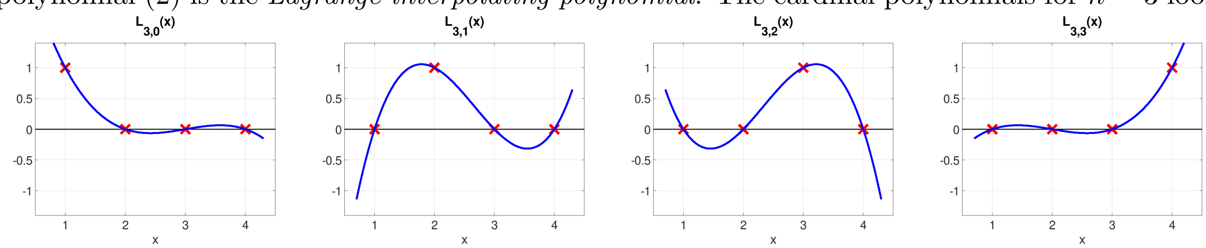
Then $L_{n,k}(x_i) = 0$ for $i = 0, \ldots, k-1, k+1, \ldots, n$ and $L_{n,k}(x_k) = 1$. Now define

$$p_n(x) = \sum_{k=0}^{n} f_k L_{n,k}(x) \in \Pi_n. \tag{2}$$

Now this implies that

$$p_n(x_i) = \sum_{k=0}^{n} f_k L_{n,k}(x_i) = f_i, \quad \text{for} \quad i = 0, 1, \ldots, n. \qquad \square$$

The polynomial (2) is the *Lagrange interpolating polynomial*. The cardinal polynomials for $n = 3$ look like:



Theorem: The interpolating polynomial of degree $\leq n$ (through $n+1$ points) is unique.

Proof "One root too many". Consider two interpolating polynomials p_n, $q_n \in \Pi_n$. Difference $d_n = p_n - q_n \in \Pi_n$ satisfies $d_n(x_k) = 0$ for $k = 0, 1, \ldots, n$, i.e., d_n is a polynomial of degree at most n but has at least $n+1$ distinct roots. Fundamemtal Theorem of Algebra $\implies d_n \equiv 0 \implies p_n = q_n$. $\square$

Demos: `demo_03_lagrange.m` and `demo_03_lagrange_construct.m`.

Data from a smooth function Suppose that $f(x)$ has at least $n+1$ smooth derivatives in the interval (x_0, x_n). Let $f_k = f(x_k)$ for $k = 0, 1, \ldots, n$, and let p_n be the Lagrange interpolating polynomial for the data (x_k, f_k), $k = 0, 1, \ldots, n$.

Error: how large can the error $f(x) - p_n(x)$ be on the interval $[x_0, x_n]$?

Theorem: For every $x \in [x_0, x_n]$ there exists $\xi = \xi(x) \in (x_0, x_n)$ such that

$$e(x) := f(x) - p_n(x) = (x-x_0)(x-x_1)\cdots(x-x_n)\frac{f^{(n+1)}(\xi)}{(n+1)!},$$

where $f^{(n+1)}$ is the $(n+1)$-st derivative of f.

Proof (sketch) Trivial for $x = x_k$, $k = 0, 1, \ldots, n$ because $e(x) = 0$ by construction. So suppose $x \neq x_k$. Key idea is to define

$$\phi(t) := e(t) - \frac{e(x)}{\pi(x)}\pi(t),$$

[1] Some material adapted from Ch. 6 of the numerical analysis textbook by Süli and Mayers.

where $\pi(t) := (t-x_0)(t-x_1)\cdots(t-x_n) = t^{n+1} + \cdots \in \Pi_{n+1}$. Now count at how many points ϕ vanishes. Note ϕ' must vanish at one fewer points. Continue recursively until there is only a single root.

Error for equispaced points Common in practice to want to use an equispaced grid with spacing $h = \frac{x_n - x_0}{n}$. In this case the error formula becomes

$$|f(x) - p_n(x)| \le \frac{h^{n+1}}{4(n+1)} \max_{\xi \in [x_0, x_n]} f^{(n+1)}(\xi).$$

But be careful! There are two terms there: how do they balance as n gets large?

Runge phenomenon Famous example due to Carl Runge (1901); the error from the interpolating polynomial approx. to $f(x) = (1+x^2)^{-1}$ for $n+1$ equally-spaced points on $[-5, 5]$ diverges near ± 5 as n tends to infinity: try `demo_03_runge.m`.

Practical advice So polynomial interpolation in a large number of equispaced points is a bad idea! Exception: periodic functions. Instead, if you can control the grid, use special "clustered" points such as Chebyshev points (see Runge demo and "Chebfun" software). Or use smaller number of points ($n \le 8$), in a local piecewise fashion: e.g., `interp` in MATLAB/Octave, also "splines".

For uniform grids of samples (e.g., from `meshgrid`), polynomial interpolation can be applied in a dimension-by-dimension fashion (`interp2`/`interp3`). For scattered data, see e.g., Radial Basis Functions.

Computer implementation The *Barycentric Formula for Lagrange Interpolation* gives a more stable and cheaper way of implementing interpolation (compared to working with the cardinal polynomials directly):

$$p_n(x) = \frac{\displaystyle\sum_{k=0}^{n} \frac{w_k}{(x-x_k)} f(x_k)}{\displaystyle\sum_{k=0}^{n} \frac{w_k}{(x-x_k)}}, \tag{3}$$

where $w_k = \frac{1}{(x_k - x_0)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_n)}$. Note: can still suffer from Runge phenomenon.

Building Lagrange interpolating polynomials from lower degree ones

Notation: let $Q_{i,j}$ be the interpolating polynomial at x_k, $k = i, \ldots, j$.

Theorem:

$$Q_{i,j}(x) = \frac{(x-x_i)Q_{i+1,j}(x) - (x-x_j)Q_{i,j-1}(x)}{x_j - x_i} \tag{4}$$

Proof: Let $s(x)$ denote the right-hand side of (4). Because of uniqueness, we wish to show that $s(x_k) = f_k$ and that the $s(x)$ is of the correct degree; left as exercises. $\square$

This can be used as the basis for constructing interpolating polynomials. In many textbooks (e.g., Burden and Faires) you'll find topics such as the Newton form and divided differences.

Generalization: Hermite interpolation Given data f_i and g_i at distinct x_i, $i = 0, 1, \ldots, n$, with $x_0 < x_1 < \cdots < x_n$, can we find a polynomial p such that $p(x_i) = f_i$ and $p'(x_i) = g_i$? Yes, there is such a unique polynomial $p_{2n+1} \in \Pi_{2n+1}$:

Construction: given $L_{n,k}(x)$ in (1), let $H_{n,k}(x) = [L_{n,k}(x)]^2(1 - 2(x-x_k)L'_{n,k}(x_k))$ and $K_{n,k}(x) = [L_{n,k}(x)]^2(x - x_k)$. Then the *Hermite interpolating polynomial*

$$p_{2n+1}(x) = \sum_{k=0}^{n} [f_k H_{n,k}(x) + g_k K_{n,k}(x)] \tag{5}$$

interpolates the data as required.

What's next? We can use $p_n(x)$ as a *proxy* for the (unknown) $f(x)$: numerical integration/differentiation.