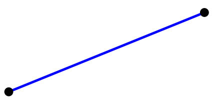
Lecture 1: Lagrange Interpolation

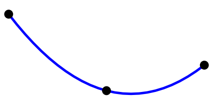
This lecture adapted from the numerical analysis textbook by Süli and Mayers, Ch. 6.

Notation: Π_n = real polynomials of degree $\leq n$

Setup: given data f_i at distinct x_i, $i = 0, 1, ..., n$, with $x_0 < x_1 < ... < x_n$, can we find a polynomial p_n such that $p_n(x_i) = f_i$? Such a polynomial is said to **interpolate** the data.

E.G.: constant $n = 0$ linear $n = 1$ quadratic $n = 2$





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

Proof: Consider, for $k = 0, 1, ..., n$, the "cardinal polynomial"

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

Then

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

So now define

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

therefore

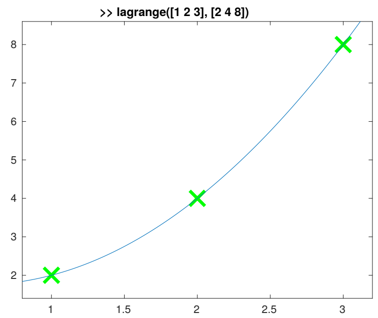
$$p_n(x_i) = \sum_{k=0}^{n} f_k L_{n,k}(x_i) = f_i \text{ for } i = 0, 1, ..., n.$$

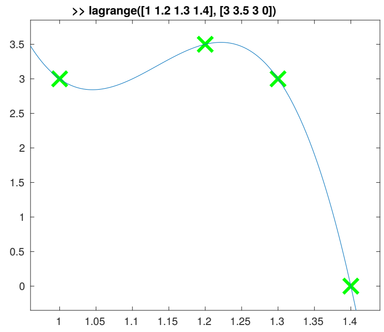
The polynomial is the **Lagrange interpolating polynomial**.

Theorem. The interpolating polynomial of degree $\leq n$ is unique.

Proof. Consider two interpolating polynomials $p_n, q_n \in \Pi_n$. Their difference $d_n = p_n - q_n \in \Pi_n$ satisfies $d_n(x_k) = 0$ for $k = 0, 1, ..., n$. i.e., d_n is a polynomial of degree at most n but has at least $n + 1$ distinct roots. Therefore, $d_n \equiv 0$, and $p_n = q_n$.

Matlab: [`lagrange.m`]





Data from an underlying 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 $k = 0, 1, ..., n$, and let p_n be the Lagrange interpolating polynomial for the data (x_k, f_k), $k = 0, 1, ..., n$.

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

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

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

Proof. Trivial for $x = x_k$, $k = 0, 1, ..., n$ as $e(x) = 0$. So suppose $x \neq x_k$. Let

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

where

$$\pi(t) \equiv (t - x_0)(t - x_1)...(t - x_n) = t^{n+1} + \text{L.O.T.} \in \Pi_{n+1}.$$

Now note that ϕ vanishes at $n+2$ points x and x_k, $k = 0, 1, ..., n$. Therefore ϕ' vanishes at $n+1$ points $\xi_0, ..., \xi_n$ between these points. Therefore ϕ'' vanishes at n points between these new points, and so on until $\phi^{(n+1)}$ vanishes at an (unknown) point ξ in (x_0, x_n). But

$$\phi^{(n+1)}(t) = e^{(n+1)}(t) - \frac{e(x)}{\pi(x)}\pi^{(n+1)}(t) = f^{(n+1)}(t) - \frac{e(x)}{\pi(x)}(n+1)!$$

since $p_n^{(n+1)}(t) \equiv 0$ and because $\pi(t)$ is is a monic polynomial of degree $n+1$. The result then follows immediately from this identity since $\phi^{(n+1)}(\xi) = 0$.

Example: $f(x) = \log(1+x)$ on $[0, 1]$. Here, $|f^{(n+1)}(\xi)| = n!/(1+\xi)^{n+1} < n!$ on $(0, 1)$. So $|e(x)| < |\pi(x)|n!/(n+1)! \leq 1/(n+1)$ since $|x - x_k| \leq 1$ for each $x, x_k, k = 0, 1, ..., n$, in $[0, 1]$, therefore $|\pi(x)| \leq 1$. This is probably pessimistic for many x, e.g. for for $x = \frac{1}{2}, \pi(\frac{1}{2}) \leq 2^{-(n+1)}$ as $|\frac{1}{2} - x_k| \leq \frac{1}{2}$.

This shows the important fact that when using equally-spaced points, the error can be large at the end points, an effect known as the "Runge phenomena" (Carl Runge, 1901). Try demo `lec01 runge.m`.

Building Lagrange interpolating polynomials from lower degree ones.

Theorem. Let $Q_{i,j}$ be the Lagrange interpolating polynomial at x_k, $k = i, ..., j$. Then:

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

Proof. Because of uniqueness, we simply wish to show the RHS interpolates the given data…

Comment: this can be used as the basis for constructing interpolating polynomials. In textbooks, often find topics such as the Newton form and divided differences.

Generalisation: Hermite interpolating polynomial matches function data and derivative data. Can also be constructed in terms of $L_{n,k}$.