* Lagrange Interpolating Polynomials

Choose the interpolating polynomial in a special form:

$$P_n(x) = \sum_{k=0}^{n} \phi_k(x) y_k \text{, where}$$

$$(*) \quad \phi_k(x) = \frac{\prod_{i=0,\, i\neq k}^{n} (x - x_i)}{\prod_{\substack{i=0 \\ i\neq k}}^{n} (x_k - x_i)}$$

Lagrange / cardinal polynomial

(the terms $i = k$ are omitted both in the numerator and denominator)

For example (with $n = 3$)

$$P_3(x) = \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} y_0 + \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} y_1$$

$$+ \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} y_2 + \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} y_3$$

Remarks

- the interpolating polynomial is a sum of $n+1$ n-degree polynomials (Lagrange functions)
- the Lagrange functions have the property

$$\phi_k(x_i) = \delta_{ik}, \quad i,k = 0,\ldots,n,$$

i.e., have unit value at the corresponding node and zero at <u>all</u> other nodes

Thanks to this property, the interpolation conditions

$$P_n(x_i) = y_i \quad , i = 0, \dots, n$$

are satisfied automatically (by construction).

— the interpolating polynomial is easy to construct (no need to solve an algebraic system)

— computational cost:

$$\left\{ \begin{array}{l} O(n) \text{ operations to evaluate} \\ \ell_i(x) \text{ for each } x \end{array} \right\} \times n = O(n^2)$$

Can we do better than that?

Examples:
- lagrange_poly.m
- lagrange_interpolation.m

Barycentric Interpolation Formula

(Chapt. 5 in Trefethen)

For all k, the numerators in (*) are the same, except that they miss different factors $(x - x_k)$. One should take advantage of that

Define the nodal polynomial for the given grid

$$\ell(x) = \prod_{i=0}^{n} (x - x_k)$$

Then, the Lagrange polynomials become

$$\ell_k(x) = \frac{\ell(x)}{\ell'(x_k)(x - x_k)}$$

We can also define

$$\lambda_k = \frac{1}{\prod_{\substack{i=0 \\ i \neq k}}^{n} (x_k - x_i)} = \frac{1}{\ell'(x_k)}$$

so that we have $\ell_k = \ell(x) \dfrac{\lambda_k}{x - x_k}$, $k = 0, \ldots, n$

Then, the Lagrange interpolation formula becomes

$$p_n(x) = \ell(x) \sum_{k=0}^{n} \frac{\lambda_k}{x - x_k} y_k$$

The First Form of the Barycentric Interpolation formula

Remarks

- Simple dependence on x inside the sum
- If the weights λ_k are known, the formula produces a value for $p(x)$ in just $O(n)$ operations
- evaluation of the weights λ_k requires $O(n^2)$ computation, but this is independent of x and hence can be performed just once at the beginning (for special grids the weights are known analytically)

- The barycentric formula is numerically stable (w.r.t round-off errors)

- Further modifications exist, e.g., for Chebyshev interpolation.

Analysis of Errors of Polynomial Interpolation

Assume the data is obtained using the function $f(x)$ as $y_i = f(x_i)$, $i = 0, \dots, n$

Consider the error function: $E(x) = f(x) - p_n(x)$

It vanishes at $x_0, x_1, \dots, x_n$, thus

$$E(x) = f(x) - p_n(x) = (x-x_0)(x-x_1)\dots(x-x_n)\,g(x)$$

$g(x)$ accounts for the behavior between the nodes

Then

$$f(x) - p_n(x) - E(x) = f(x) - p_n(x) - (x-x_0)(x-x_1)\dots(x-x_n)\,g(x) = 0$$

Need to determine $g(x)$. For this purpose we will define a new function $W(t)$ (depending on a new variable t)

$$W(t) = f(t) - p_n(t) - (t-x_0)(t-x_1)\dots(t-x_n)\,g(x)$$

$W(t)$ has $n+2$ roots: $t = x_0, x_1, \dots, x_n$, and $t = x$.

Assume $W(t)$ is continuous & differentiable

Mean Value Theorem – there is a root of the derivative $W'(t)$ between every two roots of $W(t)$; thus, altogether, there are $n+1$ roots of the derivative

$w''(t)$ — n roots

$w'''(t)$ — $(n-1)$ roots

$\vdots$

$w^{(n+1)}(t)$ — 1 root (denoted ξ) in the interval bounded by $\{x_0, x_n, x\}$

Thus,

$$w^{(n+1)}(\xi) = 0 = \frac{d^{(n+1)}}{dt^{(n+1)}} \Big[f(t) - p_n(t) - (t-x_0)(t-x_1)\cdots(t-x_n)\, g(x) \Big]\Big|_{t=\xi}$$

$$= f^{(n+1)}(\xi) - 0 - (n+1)!\, g(x) = 0 \Rightarrow$$

$$\Rightarrow g(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}, \text{ where } \xi \text{ varies between } \{x_0, x_1, x\}$$

Therefore

$$E(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{k=0}^{n} (x - x_k)$$

Remarks

* since $f(x)$ is usually unknown, the expression for $E(x)$ has only qualitative meaning
* if $f(x) = p_m(x)$, $m \le n$, then $E(x) \equiv 0$
* smooth functions $f(x) \Rightarrow$ smaller errors
* for large n the error $E(x)$ increases at the endpoints of the interval (Runge phenomenon); do not take n larger than 5, 6, ..

Another Look at the Errors of Polynomial Interpolation

$$E_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \underbrace{\prod_{k=0}^{n} (x - x_k)}_{\omega_{n+1}(x)}$$

Assume $x \in [a,b]$

Then $x_k = a + kh$, $k = 0, \dots, n$, $h = \frac{b-a}{n}$

Let $M_n = \max_{a \le x \le b} |f^{(n)}(x)|$ — upper bound on the n-th derivative

$$c_n = \max_{0 \le z \le n} \prod_{k=0}^{n} |z - k|$$

Thus: $E_{max} = \max_{a \le x \le b} |E_n(x)| \le \frac{1}{(n+1)!} c_n M_{n+1} h^{n+1}$

For a fixed degree of interpolating polynomial $E_{max} \le c h^{n+1}$, $h \ll 1$

How do reduce the error: $h \to 0$

— shrink the interval $|b-a| \to 0$, while keeping n unchanged

— keep the interval, but increase n then we need to control the derivatives $\lim_{n \to \infty} M_n$

The rate of decay (increase) of M_n depends the analytical (regularity) properties of the interpolation function

* Runge's function (1901) $f(x) = \frac{1}{1 + 25x^2}$, $x \in [-1,1]$