Numerical Methods I: Polynomial Interpolation

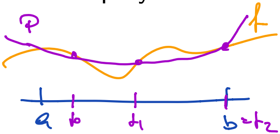
Georg Stadler
Courant Institute, NYU
stadler@cims.nyu.edu

November 16, 2017

Classical polynomial interpolation

Given $f_i := f(t_i)$, $i = 0, \dots, n$, we would like to find a polynomial $P \in \boldsymbol{P}_n$ such that

$$P(t_i) = f_i.$$



Interpolation is thus a map from $\mathbb{R}^{n+1} \to \boldsymbol{P}_n$.

Theorem: Given nodes (t_i, f_i), $0 \le i \le n$, with pairwise distinct nodes t_i, then there exists a unique interpolating polynomial $P \in \boldsymbol{P}_n$.

Proof: dim $(\mathbb{R}^{n+1}) = n+1$, dim $(\mathbb{P}_n) = n+1$,

map $\Phi : \begin{pmatrix} f^0 \\ \vdots \\ f^n \end{pmatrix} \longrightarrow \mathbb{P}_n$ interpolating polynomial, Sufficient to show that Φ is injective: Let $f, g \in \mathbb{R}^{n+1}$: $\Phi(f) = \Phi(g)$

$\Phi(f)(t_i) = \Phi(g)(t_i)$

$\Longrightarrow \Phi(f-g) = 0$, $\Phi(f-g)$ is zero polynomial, $\Longrightarrow f_i = g_i$ $i = 0, \dots, n$ $\Longrightarrow f = g$

To compute that polynomial, we have to choose a basis in $\boldsymbol{P}_n$.

$$\Phi(f+g) = \Phi(f) + \Phi(g)$$
$$\Phi(\alpha f) = \alpha \Phi(f)$$

Classical polynomial interpolation

Monomial basis: $1, t, t^2 \ldots$ leads to system with Vandermonde matrix V_n.

$p(t) = a_0 + a_1 t + \ldots + a_n t^n$, we want $p(t_i) = f_i$

$\Longrightarrow \quad a_0 + a_1 t_i + \ldots + a_n t_i^n = f_i \qquad i = 0, \ldots, n$

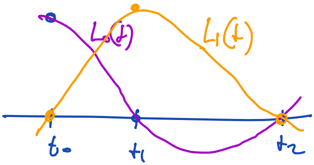
$$\underbrace{\begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^n \\ \vdots & & & & \vdots \\ 1 & t_n & t_n^2 & \cdots & t_n^n \end{bmatrix}}_{V_n \ldots \text{Vandermonde matrix for } t_0, \ldots, t_n} \begin{bmatrix} a_0 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ \vdots \\ f_n \end{bmatrix}$$

- $det(V_n) = \prod_{i=0}^{n} \prod_{j=i+1}^{n} (t_i - t_j) \neq 0$
- For larger n, this can be a poorly conditioned system.

Classical polynomial interpolation

Lagrange basis L_i defined by $L_i(t_j) = \delta_{ij}$.

$$L_i(t) = \frac{\prod_{j\neq i}(t-t_j)}{\prod_{j\neq i}(t_i-t_j)}$$



- Simple interpolant: $P(t) = \sum_{i=0}^{n} f_i L_i(t)$
- Not always practical.
- Lagrange polynomials form an orthogonal basis in $\boldsymbol{P}_n$ w.r. to the inner product

$$(P, Q) := \sum_{i=0}^{n} P(t_i)Q(t_i) = \int_a^b W(t)P(t)Q(t)\,dt$$

$$W(t) = \sum_{i=0}^{n} \delta_{t=t_i} \quad \text{(Dirac delta functions)}$$

Classical polynomial interpolation

The Newton basis $\omega_0, \ldots, \omega_n$ is given by

$$\omega_i(t) := \prod_{j=0}^{i-1} (t - t_j) \in \boldsymbol{P}_i.$$

This polynomials are linearly independent as their degree increases.

$w_0(t) = 1$, $w_1(t) = (t - t_0)$, $w_2(t) = (t-t_0)(t-t_1)$, ...

polynomial interpolation $a_0 w_0(t) + a_1 w_1(t) + \ldots + a_n w_n(t)$
and there's an efficient way to compute $a_0, a_1, \ldots a_n$.

The coefficients in this basis can be computed efficiently (more later).

Interpolation can be seen as map between

$$\bar{\Phi} : \mathbb{R}^{n+1} \mapsto \boldsymbol{P}_n$$

or as map between functions:

$$\Phi : C([a, b]) \mapsto \boldsymbol{P}_n.$$

Φ is function evaluation at the nodes, followed by $\bar{\Phi}$.

Theorem: Let $a \leq t_0 < \ldots < t_n \leq b$ be pairwise distinct and L_i be the corresponding Lagrange polynomials. Then the absolute condition number of the polynomial interpolation:

$$\Phi : C([a,b]) \to \boldsymbol{P}_n$$

Supremum norm of f is : $\|f\|_\infty = \sup_{x \in [a,b]} |f(x)|$

w.r. to the supremum norm is the Lebesgue constant

$$\kappa_{\mathsf{abs}} = \Lambda_n = \max_{t \in [a,b]} \sum_{i=1}^{n} |L_i(t)|.$$

Note that the Lebesgue constant depends on n and the location of the t_i.

Classical polynomial interpolation

Conditioning

$f \in C([a,b])$

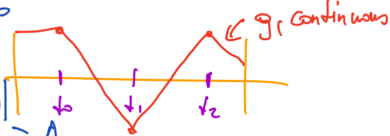
<u>Proof:</u> $t \in [a,b]$: $\quad |\Phi(f)(t)| = \left| \sum_{i=0}^{n} f(t_i) L_i(t) \right|$

$$\le \sum_{i=0}^{n} |f(t_i)|\,|L_i(t)| \le \|f\|_\infty \underbrace{\sum_{i=0}^{n} |L_i(t)|}_{\le \max_{t\in[a,b]} \sum_{i=0}^{n} |L_i(t)| = \Lambda_n}$$

$$\Longrightarrow \|\Phi(f)\|_\infty \le \|f\|_\infty \Lambda_n$$

$$\Longrightarrow \kappa_{abs} \le \Lambda_n$$

Show that this estimate is sharp: $g \in C([a,b])$, target is to find τ with $\Phi(g)(\tau) = \|g\|_\infty \Lambda_n$; choose τ where the maximum in the Lebesgue constant is obtained, i.e. $\sum_{i=0}^{n} |L_i(\tau)| = \Lambda_n$, choose g with $\|g\|_\infty = 1$, $g(t_i) = \operatorname{sgn} L_i(\tau)$



$$\Longrightarrow |\Phi(g)(\tau)| = \left| \sum_{i=0}^{n} g(t_i) L_i(\tau) \right| = \sum_{i=0}^{n} |L_i(\tau)| = \Lambda_n$$

$$\Longrightarrow \kappa_{abs} \ge \Lambda_n \Longrightarrow \boxed{\kappa_{abs} = \Lambda_n}$$

Classical polynomial interpolation

Conditioning

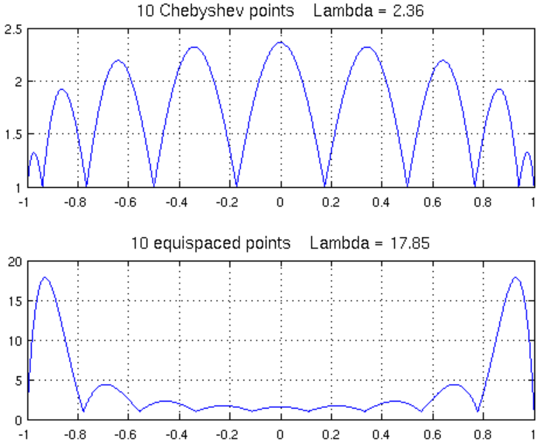
Lebesgue constants for different orders:

n	Λ_n for equidistant nodes	Λ_n for Chebyshev nodes
5	3.106292	2.104398
10	29.890695	2.489430
15	512.052451	2.727778
20	10986.533993	2.900825

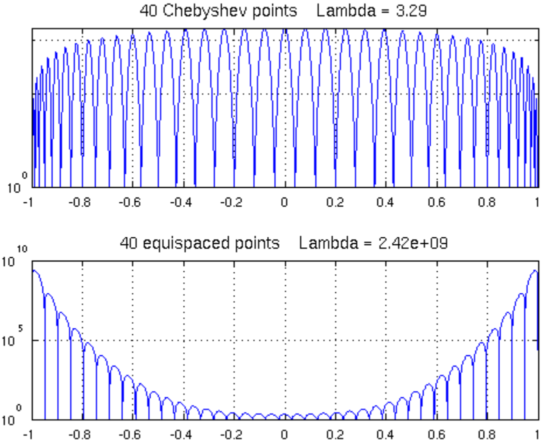
Chebyshev nodes are the roots of the Chebyshev polynomials:

$$t_i = \cos\left(\frac{2i+1}{2n+2}\pi\right), \text{ for } i = 0, \ldots, n$$

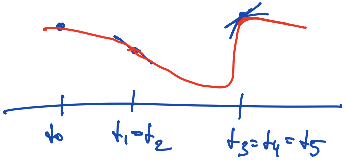
Lebesgue constant for $n = 10$, uniform vs. Chebyshev nodes:



Lebesgue constant for $n = 40$, uniform vs. Chebyshev nodes:



Hermite interpolation



Assume

$$a = t_0 \leq t_1 \leq \ldots \leq t_n = b$$

with possibly duplicated nodes. If the node t_i occurs k times, the corresponding node values correspond to $f(t_i), f'(t_i), \ldots, f^{k-1}(t_i)$.

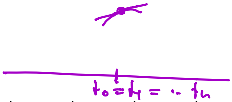
The Hermite interpolation polynomial $p(x)$ is a polynomial of order n, which coincides with the nodal values (and, for duplicated nodes, derivatives at nodal values) at the nodes.

Hermite interpolation

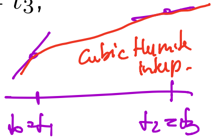
Theorem: (somewhat loosely formulated version) Given $n+1$ nodes and nodal values (possibly of derivatives), then there exists a unique interpolating Hermite polynomial $p \in \boldsymbol{P}_n$.

Examples:

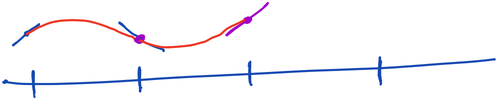
- All $t_0 = \ldots = t_n$.



- Cubic Hermite interpolation: Nodes: $t_0 = t_1 < t_2 = t_3$, Values: $f(t_0), f'(t_0), f(t_1), f'(t_1)$.



- locally cubic Hermite interpolation.



cubic Hermite on each interval

The Newton basis $\omega_0, \dots, \omega_n$ is given by

$$\omega_i(t) := \prod_{j=0}^{i-1} (t - t_j) \in \boldsymbol{P}_i.$$

The leading coefficient a_n of the interpolation polynomial of f

$$P(f|t_0, \dots, t_n) = a_n x^n + \dots$$

is called the n*-th divided difference*, $[t_0, \dots, t_n]f := a_n$.

Theorem: For $f \in C^n$, the interpolation polynomial $P(f|t_0, \dots, t_n)$ is given by

$$P(t) = \sum_{i=0}^{n} [t_0, \dots, t_i] f \, \omega_i(t).$$

If $f \in C^{n+1}$, then

$$f(t) = P(t) + [t_0, \dots, t_n, t] f \; \omega_{n+1}(t).$$

This property allows to estimate the interpolation error.

Proof: $n=0$ ✓ Let $n>0$, true for $n-1$, i.e.

$$P_{n-1} = P(f|t_0,\dots,t_{n-1}) = \sum_{i=0}^{n-1} \underbrace{[t_0,\dots,t_i]f}_{\in \mathbb{R},\ i\text{-th divided difference}}\, w_i(t)$$

$$P_n = P(f|t_0,\dots,t_n) = [t_0,\dots,t_n]f\, t^n + \dots$$

$$= [t_0,\dots,t_n]f\, w_n(t) + Q_{n-1}(t)$$

$\in \mathbb{P}_{n-1}$

$$Q_{n-1}(t) = P_n - [t_0,\dots,t_n]f\, w_n(t)$$

interpolates f at $t_0,\dots,t_{n-1}$ because $P(t_i)=f_i$

$w_n(t_i)=0$

$$\Longrightarrow Q_{n-1} = P_{n-1}$$

use result for $n-1$ $\Longrightarrow$ $P_n = [t_0,\dots,t_n]f\, w_n(t) + \sum_{i=1}^{n-1} [t_0,\dots,t_i]f\, w_i(t) \Longrightarrow$ ✓

remaining claim: use that

$P_n(t) + [t_0,\dots,t_n,t]f\, w_{n+1}$ interpolates at $t_0,\dots,t_n,t$

The divided differences $[t_0, \ldots, t_n]f$ satisfy the following properties:

- $[t_0, \ldots, t_n]P = 0$ for all $P \in \boldsymbol{P}_{n-1}$.

  (def. of divided differences)

- If $t_0 = \ldots = t_n$: (Taylor expansion)

$$[t_0, \ldots, t_n]f = \frac{f^{(n)}(t_0)}{n!}$$

nodes.

- The following recurrence relation holds for $t_i \neq t_j$ (nodes with a hat are removed):

$$[t_0, \dots, t_n]f = \frac{\left([t_0, \dots, \hat{t_i}, \dots, t_n]f - [t_0, \dots, \hat{t_j}, \dots, t_n]f\right)}{t_j - t_i}$$

- If $f \in C^n$ $[t_0, \dots, t_n]f = \frac{1}{n!} f^{(n)}(\tau)$ with an $a \leq \tau \leq b$, and the divided differences depend continuously on the nodes.

Let us use divided differences to compute the coefficients for the Newton basis for the cubic interpolation polynomial p that satisfies $p(0) = 1$, $p(0.5) = 2$, $p(1) = 0$, $p(2) = 3$.

t_i				
0	$[t_0]f = 1$			
0.5	$[t_1]f = 2$	$[t_0 t_1]f = \frac{[t_1]f - [t_0]f}{t_1 - t_0} = 2 = \frac{2-1}{0.5-0} = 2$		
1	$[t_2]f = 0$	$[t_1 t_2]f = \frac{[t_2]f - [t_1]f}{t_2 - t_1} = -4$	$[t_0 t_1 t_2]f = -6 = \frac{-4-2}{1-0} = -6$	
2	$[t_3]f = 3$	$[t_2 t_3]f = \frac{[t_3]f - [t_2]f}{t_3 - t_2} = 3$	$[t_1 t_2 t_3]f = \frac{14}{3}$	$\frac{16}{3}$

Thus, the interpolating polynomial is

$$p(t) = 1 + 2t + (-6)t(t - 0.5) + \frac{16}{3}t(t - 0.5)(t - 1).$$

Classical polynomial interpolation

Divided differences

Let us now use divided differences to compute the coefficients for the Newton basis for the cubic interpolation polynomial p that satisfies $p(0) = 1$, $p'(0) = 2$, $p''(0) = 1$, $p(1) = 3$.

t_i				
0	$[t_0]f = 1$			
0	$[t_0]f = 1$	$[t_0 t_1]f = p'(0) = 2$		
0	$[t_0]f = 1$	$[t_1 t_2]f = p'(0) = 2$	$[t_0 t_1 t_2]f = \frac{p''(0)}{2!} = \frac{1}{2}$	
1	$[t_3]f = 3$	$[t_2 t_3]f = \frac{[t_3]f - [t_0]f}{t_3 - t_0} = 2$	0	$-\frac{1}{2}$

Thus, the interpolating polynomial is

$$p(t) = 1 + 2t + \frac{1}{2}t^2 + (-\frac{1}{2})t^3$$

$w_1(t)$ $w_2(t)$ $w_3(t)$

Classical polynomial interpolation

Approximation error

If $f \in C^{(n+1)}$, then

$$f(t) - P(f|t_0, \ldots, t_n)(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!}\omega_{n+1}(t)$$

for an appropriate $\tau = \tau(t)$, $a < \tau < b$.

In particular, the error depends on the choice of the nodes.

For Taylor interpolation, i.e., $t_0 = \ldots = t_n$, this results in:

$$f(t) - P(f|t_0, \ldots, t_n)(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!}(t - t_0)^{n+1}$$

Consider functions

$$\{f \in C^{n+1}([a,b]) : \sup_{\tau \in [a,b]} |f^{n+1}(\tau)| \leq M(n+1)!\}$$

for some $M > 0$, then the approximation error depends on $\omega_n(t)$, and thus on $t_0, \ldots, t_n$.

Thus, one can try to minimize

$$\max_{a \leq t \leq b} |\omega_{n+1}(t)|,$$

which is achieved by choosing the nodes as the roots of the Chebyshev polynomial of order $(n+1)$.

Summary on pointwise convergence:

- If an interpolating polynomial is close/converges to the original function depends on the regularity of the function and the choice of interpolation nodes
- For a good choice of interpolation nodes, fast convergence can be obtained for almost all functions

Classical polynomial interpolation

- Polynomial interpolation
- Least squares with polynomials
- Splines (i.e., piecewise polynomial interpolation):