Chapter 2: Polynomial Interpolation

Contents

1 Polynomial Interpolation

In this chapter we study how to interpolate a data set with a polynomial.

Problem Description: Given (n+1) points, say(x_i, y_i), where i = 0,1,2,. . . ,n, with distinct x_i, not necessarily sorted, we want to find a polynomial of degree n:

$$P_n(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x + a_0$$

such that it interpolates these points, i.e.,

$$P_n(x) = y_i, i = 0, 1, 2, ..., n$$

The goal is the determine the coefficients $a_n, a_{n-1}, ..., a_1, a_0$. Note the total number of data poitns is 1 larger than the degree of the polynomial.

Why should we do this?

- Find the values between the points for discrete data set
- To approximate a (probably complicated) function by a polynomial
- Then, it is easier to do computations such as derivative, integrations, etc. . .

1.1 Van Der Monde Matrix Example

Interpolate the given data set with a polynomial of degree 2:

x_i	0	1	2/3
y_i	1	0	0.5

Answer Let

$$P_2(x) = a_2x^2 + a_1x + a_0$$

We need to find the coefficients a_2, a_1, a_0. By the interpolating properties, we have 3 equations:

$$x = 0, y = 1 : P_2(0) = a_0 = 1$$
$$x = 1, y = 0 : P_2(1) = a_2 + a_1 + a_0 = 0$$
$$x = \frac{2}{3}, y = 0.5 : P_2(\frac{2}{3}) = (\frac{4}{9})a_2 + (\frac{2}{3})a_1 + a_0 = 0.5$$

Here we have 3 linear equations and 3 unknowns (a_2, a_1, a_0).

The equations:

$$a_0 = 1$$
$$a_2 + a_1 = 0$$
$$\frac{4}{9}a_2 + \frac{2}{3}a_1 + a_0 = 0.5$$

In matrix-vector form:

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ \frac{4}{9} & \frac{2}{3} & 1 \end{pmatrix} \begin{pmatrix} a_2 \\ a_1 \\ a_0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0.5 \end{pmatrix}$$

Easy to solve in Matlab, or do it by hand:

$$a_2 = -\frac{3}{4}$$
$$a_1 = -\frac{1}{4}$$
$$a_0 = 1$$

Then

$$P_2(x) = -\frac{3}{4}x^2 - \frac{1}{4}x + 1$$

The general case. For the general case with $(n+1)$ points, we have

$$P_n(x_i) = y_i, \qquad i = 0, 1, 2, \cdots, n$$

We will have $(n+1)$ equations and $(n+1)$ unknowns:

$$\begin{aligned}
P_n(x_0) = y_0 \quad &: \quad x_0^n a_n + x_0^{n-1} a_{n-1} + \cdots + x_0 a_1 + a_0 = y_0 \\
P_n(x_1) = y_1 \quad &: \quad x_1^n a_n + x_1^{n-1} a_{n-1} + \cdots + x_1 a_1 + a_0 = y_1 \\
&\vdots \\
P_n(x_n) = y_n \quad &: \quad x_n^n a_n + x_n^{n-1} a_{n-1} + \cdots + x_n a_1 + a_0 = y_n
\end{aligned}$$

Putting this in matrix-vector form

$$\begin{pmatrix} x_0^n & x_0^{n-1} & \cdots & x_0 & 1 \\ x_1^n & x_1^{n-1} & \cdots & x_1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_n^n & x_n^{n-1} & \cdots & x_n & 1 \end{pmatrix} \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

i.e.

$$\mathbf{X}\,\vec{a} = \vec{y}$$

Figure 1:

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1 & , \quad i = j \\ 0 & , \quad i \neq j \end{cases} \qquad i = 0, 1, \cdots, n$$

Figure 2:

1.1.1 The general case

- X: a (n+1) x (n+1) matrix, given (Van Der Monde matrix)
- a: unknown vector, with length (n + 1)
- y: given vector, with length (n + 1)

Theorem If x_i's are distinct, then X is invertible, therefore a has a unique solution.

In Matlab, the command vander(x), where x is a vector that contains the interpolation points x = $[x_1, x_2, ..., x_n]$, will generate this matrix.

Bad News: X has a very large condition number for large n, therefore not effective to solve if n is large.

1.2 Lagrange Form

Given points: $x_0, x_1, ..., x_n$

Define the **cardinal functions** $I_0, I_1, ..., I_n :\in P^n$, satisfying the properties

Here δ_{ij} is called the Kronecker's delta.

Locally supported in discrete sense. The cardinal functions I_i(x) can be written as:

Lagrange form of the interpolation polynomial can be simply expressed as

$$
\begin{aligned}
l_i(x) &= \prod_{j=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \\
&= \frac{x - x_0}{x_i - x_0} \cdot \frac{x - x_1}{x_i - x_1} \cdots \frac{x - x_{i-1}}{x_i - x_{i-1}} \cdot \frac{x - x_{i+1}}{x_i - x_{i+1}} \cdots \frac{x - x_n}{x_i - x_n}
\end{aligned}
$$

Verify:

$$l_i(x_i) = 1$$

and for $i \neq k$

$$l_i(x_k) = 0$$

$$l_i(x_k) = \delta_{ik}.$$

Figure 3:

$$P_n(x) = \Sigma I_j(x) * y_i$$

It is easy to check the interpolating property:

$$P_n(x_j) = \Sigma I_i(x_j) * y_i = y_j$$

for every j. (The cardinal function is 1 at x_i, so y_i is multiplied by 1).

Example 2 Write the Lagrange polynomial for the data

x_i	0	1	2/3
y_i	1	0	0.5

Answer We first ocmpute the cardinal functions

$$
\begin{aligned}
I_0(x) &= \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x - 2/3)(x - 1)}{(0 - 2/3)(0 - 1)} = \frac{3}{2}(x - \frac{2}{3})(x - 1) \\
I_1(x) &= \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{(x - 0)(x - 1)}{(2/3 - 0)(2/3 - 1)} = -\frac{9}{2}x(x - 1) \\
I_2(x) &= \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{(x - 0)(x - 2/3)}{(1 - 0)(1 - 2/3)} = 2x(x - \frac{2}{3})
\end{aligned}
$$

so

$$P_2(x) = I_0(x)y_0 + I_1(x)y_1 + I_2(x)y_2 = \frac{3}{2}(x-\frac{2}{3})(x-1) - \frac{9}{2}x(x-1)(0.5) + 0 = -\frac{3}{4}x^2 - \frac{1}{4} + 1$$

Pros and cons of Lagrange polynomial:

- (+) Elegant Formula
- (-) Slow to compute, since each cardinal function is different
- (-) Not flexible: if one changes a point x_j, or add on an additional point x_{n+1}, one must recompute all cardinal functions.

1.3 Newton's divided differences

Given a data set:

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

We will describe an algorithm in a recursive form.

Main idea:

Given $P_k(x)$ that interpolates k + 1 data points $\{x_i, y_i\}$, i = 0,1,2,...,k, compute $P_{k+1}(x)$ that interpolates one extra point, $\{x_{k+1}, y_{k+1}\}$, by using P_k and adding an extra term

For n = 0, we set $P_0(x) = y_0$. Then $P_0(x_0) = y_0$.

For n = 1, we set

$$P_1(x) = P_0(x) + a_1(x - x_0)$$

where a_1 is to be determined.

Then, $P_1(x_0) = P_0(x_0) + 0 = y_0$, for any a_1.

Find a_1 by the interpolation property $y_1 = P_1(x_1)$, we have

$$y_1 = P_0(x_1) + a_1(x_1 - x_0) = y + 0 + a_1(x_1 - x_0)$$

This gives us

$$a_1 = \frac{y_1 - y_0}{x_1 - x_0}$$

For n = 2, we set

$$P_2(x) = P_1(x) + a_2(x - x_0)(x - x_1)$$

Then $P_2(x_0) = P_1(x_0) = y_0, P_2(x_1) = P_1(x_1) = y_1$.

Determine a_2 by the interpolation property $y_2 = P_2(x_2)$

$$y_2 = P_1(x_2) + a_2(x_2 - x_0)(x_2 - x_1)$$

Then

$$a_2 = \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}$$

We would like to express a_2 in a different way. Recall

$$P_1(x) = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)$$

Then

$$P_1(x_2) = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x_2 - x_0)$$

$$P_1(x_2) = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x_2 - x_1) + \frac{y_1 - y_0}{x_1 - x_0}(x_1 - x_0)$$

$$P_1(x_2) = y_1 + \frac{y_1 - y_0}{x_1 - x_0}(x_2 - x_1)$$

Then a_2 can be rewritten as

$$a_2 = \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} = \frac{y_2 - y_1 - \frac{y_1 - y_0}{x_1 - x_0}(x_2 - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{\frac{y_2 - y_1}{x_2 - x_1} - \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0}$$

(Rise over the run? Secant lines represented. . .)

$$a_2 \approx \frac{f'(x_2) - f'(x_0)}{x_2 - x_0} \approx f''()$$

and a_1 is an approximate first derivative.

1.3.1 The general case:

Newton's form for the interpolation polynomial:

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + ... + a_n(x - x_0)(x - x_1)...(x - x_{n-1})$$

1.3.2 Recursive Computation

The recursion is initiated with

$$f[x_i] = y_i, i = 0, 1, 2, ...$$

Then

$$f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$$

$$f[x_1, x_2] = \frac{f[x_2] - f[x_1]}{x_2 - x_1}$$

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_1, x_0]}{x_2 - x_0}$$

The general case for a_n:
Assume that $P_{n-1}(x)$ interpolates (x_i, y_i) for $i = 0, 1, \cdots, n-1$.
Let

$$P_n(x) = P_{n-1}(x) + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

Then for $i = 0, 1, \cdots, n-1$, we have

$$P_n(x_i) = P_{n-1}(x_i) = y_i.$$

Find a_n by the property $P_n(x_n) = y_n$,

$$y_n = P_{n-1}(x_n) + a_n(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-1})$$

then

$$a_n = \frac{y_n - P_{n-1}(x_n)}{(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-1})}$$

Figure 4:

$$f[x_0, x_1, x_2] = \frac{f[x_3, x_2] - f[x_2, x_1]}{x_3 - x_1}$$

For the general step we have

$$f[x_0, x_1, ..., x_k] = \frac{f[x_1, x_2, ..., x_k] - f[x_0, x1_{.}.., x_{k-1}]}{x_k - x_0}$$

The constants a_k's in the Newton's form are computed as

$$a_0 = f[x_0]$$
$$a_1 = f[x_0, x_1]$$
$$...$$
$$a_k = f[x_0, x_1, ..., x_k]$$

We compute f[...]'s through the following table:

The diagonal elements give us the coefficients a_i's

1.3.3 Example

Write Newton's form of interpolation polynomial for the data

x_i	0	1	2/3	1/3
y_i	1	0	1/2	0.866

Answer Set up the triangular table for computation

0	1			
1	0	-1		
2/3	0.5	-1.5	-0.75	
1/3	0.8660	-1.0981	-0.6029	0.4413

x_0	$f[x_0] = y_0$				
x_1	$f[x_1] = y_1$	$f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$			
x_2	$f[x_2] = y_2$	$f[x_1, x_2] = \frac{f[x_2] - f[x_1]}{x_2 - x_1}$	$f[x_0, x_1, x_2]$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
x_n	$f[x_n] = y_n$	$f[x_{n-1}, x_n] = \frac{f[x_n] - f[x_{n-1}]}{x_n - x_{n-1}}$	$f[x_{n-2}, x_{n-1}, x_n]$	$\dots$	$f[x_0, x_1, \cdots, x_n]$

Figure 5:

So we have

$$a_0 = 1, a_1 = -1, a_2 = -0.75, a_3 = 0.4413$$

Then

$$P_3(x) = 1 + -1x + -0.75x(x-1) + 0.4413x(x-1)x - \frac{2}{3})$$

Flexibility of Newton's form: easy to add additional points to interpolate.

1.3.4 Nested form of Newton's Polynomial

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + ... + a_n(x - x_0)(x - x_1)...(x - x_{n-1})$$
$$P_n(x) = a_0 + (x - x_0)(a_1 + (x - x_1)(a_2 + (x - x_2)(a_3 + ... + a_n(x - x_{n-1}))))$$

Effective coding:

Given the data x_i and a_i for i = 0,1,. . . ,n the following pseudo-code evaluates Newton's Polynomial $p = P_n(x)$

- p = a_n
- for k = n-1,n-2,. . . ,0
  - p = p(x-x_k) + a_k
- end

This requires only 3n flops.

1.4 Existence and Uniqueness theorem for polynomial interpolation

Theorem (Fundamental Theorem of Algebra)

Every polynomial of degree n that is not identically zero, has maximum n roots (including multiplicities). These roots may be real or complex. In particular, this implies that if a polynomial of degree n has more than n roots, then it must be identically zero.

Theorem (Existence and Uniqueness of Polynomial Interpolation)

Given (x_i, y_i), with x_i's distinct. There exists one and only one polynomial P_n(x) of degree $\leq n$ such that $P_n(x_i) = y_i$ for i = 0,1,. . . ,n.

Proof

The existence: by construction. Uniqueness: Assume we have two polynomials $p(x), q(x) \in P_n$, such that

$$p(x_i) = y_i$$
$$q(x_i) = y_i$$
$$i = 0, 1, ..., n$$

Now, let $g(x) = p(x) - q(x)$, a polynomial of degree $\leq n$.

$$g(x_i) = p(x_i) - q(x_i) = y_i - y_i = 0$$
$$i = 0, 1, ..., n$$

So g(x) has n+1 zeros. By the Fundamental Theorem of Algebra (max n roots), we must have g(x) = 0, therefore p(x) is congruent to q(x).

2 Errors in Polynomial Interpolation

Given a function f(x) on $x \in [a, b]$, and a set of distinct points $x_i \in [a, b]$, i = 0,1,. . . ,n. Let $P_n(x) \in P_n$ such that..

$$P_n(x_i) = f(x_i), i = 0, 1, .., n$$

Error function

$$e(x) = f(x) - P_n(x), x \in [a, b]$$

Theorem. There exists some value $\xi \in [a, b]$, such that

$$e(x) = \frac{1}{(n+1)!}\xi\Pi(x - x_i), x \in [a, b]$$

Proof

If $f \in P_n$, then by Uniqueness Theorem of polynomial interpolation we must have $f(x) = P_n(x)$. Then e(x) == 0 and the proof is trivial.

Now assume $f \notin P_n(x)$. If $x = x_i$ for some i, we have $e(x_i) = f(x_i) - P_n(x_i) = 0$, and the results holds.

Now consider $x \neq x_i$ for any i.

$$W(x) = \Pi(x - x_i) \in P_{n+1}$$

it holds:

$$W(x_i) = 0$$
$$w(x) = x^{n+1} + ...$$
$$W^{(n+1)} = (n+1)!$$

Fix a y such that $y \in [a, b]$ and $y \neq x_i$ for any i. We define a constant

$$c = \frac{f(y) - P_n(y)}{W(y)}$$

and another function

$$\phi(x) = f(x) - P_n(x) - cW(x)$$

We find all the zeros for $\phi(x)$. We see that x_i's are zeros since

$$\phi(x_i) = f(x_i) - P_n(x_i) - cW(x_i) = 0, i = 0, 1, ..., n$$

and also y is a zero because

$$\phi(y) = f(y) - P_n(y) - cW(y) = 0$$

So, ϕ has at least (n+2) zeros.

2.1 Error fomula Example

Recall the error formula:

$$e(x) = \frac{1}{(n+1)!}\xi\Pi(x - x_i)$$

Example If n = 1, x_0 = a, x_1 = b, b > a, find an upper bound for error.

Answer Let

$$M = max|f''(x)| = ||f''(x)||_\infty$$

and observe

$$max|(x - a)(x - b)| = \frac{(b - a)^2}{4}$$

For $x \in [a, b]$, we have

$$|e(x)| = \frac{1}{2}|f''(\xi)| * |(x - a)(x - b)| \leq \frac{1}{2}||f''||_\infty \frac{(b - a)^2}{4} = \frac{1}{8}||f''||_\infty (b - a)^2$$

Error depends on the distribution of nodes x_i. If b is close to a, then we have a small error bound.

Here goes our deduction:

$$\begin{array}{rccl}
\varphi(x) \text{ has at least} & n+2 & \text{zeros on } [a,b]. & \\
\varphi'(x) \text{ has at least} & n+1 & \text{zeros on } [a,b]. & \\
\varphi''(x) \text{ has at least} & n & \text{zeros on } [a,b]. & \\
 & \vdots & & \\
\varphi^{(n+1)}(x) \text{ has at least} & 1 & \text{zero on } [a,b]. & \boxed{\text{Call it } \xi \text{ s.t. } \varphi^{(n+1)}(\xi) = 0.}
\end{array}$$

So we have

$$\varphi^{(n+1)}(\xi) = f^{(n+1)}(\xi) - 0 - cW^{(n+1)}(\xi) = 0.$$

Recall $W^{(n+1)} = (n+1)!$, we have, for every y,

$$f^{(n+1)}(\xi) = cW^{(n+1)}(\xi) = \frac{f(y) - P_n(y)}{W(y)}(n+1)!.$$

Writing y into x, we get

$$e(x) = f(x) - P_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi) W(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi) \prod_{i=0}^{n} (x - x_i),$$

for some $\xi \in [a,b]$.

Figure 6:

3 Unifrom Grid

Equally distribute the nodes (x_i): on [a,b], with n+1 nodes.

$$x_i = a + ih, h = \frac{b-a}{n}, i = 0, 1, ..., n$$

One can show that for $x \in [a, b]$, it holds

$$\Pi|x - x_i| \leq \frac{1}{4}h^{n+1} * n!$$

Proof If x = x_i for some i, then x-x_i = 0 and the product is zero, so it trivially holds.

Now assume x_i < x < x_{i+1}. We have

$$max|(x - x_i)(x - x_{i+1}| = \frac{1}{4}(x_{i+1} - x_i)^2 = \frac{h^2}{4}$$

Now consider the other terms in the product, say x-x_j, for either j > i + 1, or j < i. Then $|(x-x)j| \leq h(j-i)$ for j > i + 1 and $|(x-x)j| \leq h(i+1-j)$ for j < i. In all cases, the product of these terms are bounded by $h^{n-1}n!$, proving the result.

We have the error estimate

$$|e(x)| \leq \frac{1}{4(n+1)}|f^{(n+1)}(x)|h^{n+1} \leq \frac{M_{n+1}}{4(n+1)}h^{n+1}$$

where

$$M_{n+1} = max|f^{(n+1)}(x)| = ||f^{(n+1)}||_\infty$$

Example Consider interpolating f(x) = sin(πx) with polynomial on the interval [-1, 1] with uniform noes. Give an upper bound for error.

Answer Since

$$f'(x) = \pi cos\pi x$$
$$f''(x) = -\pi^2 sin\pi x$$
$$f'''(x) = -pi^3 cos\pi x$$

we have

$$|f^{(n+1)}(x)| \leq \pi^{n+1}, M_{n+1} = \pi^{n+1}$$

so the upper bound for error is

$$|e(x)| \leq \frac{M_{n+1}}{4(n+1)}h^{n+1} \leq \frac{\pi^{n+1}}{4(n+1)}(\frac{2}{n})^{n+1}$$

Simulation Data:

n	error bound	measured error
4	4.8×10^{-1}	1.8×10^{-1}
8	3.2×10^{-3}	1.2×10^{-3}
16	1.8×10^{-9}	6.6×10^{-10}

Figure 7:

$$\max_{a \le x \le b} \left\{ \prod_{k=0}^{n} |x - \bar{x}_k| \right\} = 2^{-n} \le \max_{a \le x \le b} \left\{ \prod_{k=0}^{n} |x - x_k| \right\}$$

Figure 8:

4 Chebychev nodes: equally distributing the error

Type I: including the end points.

For interval [-1,1]: $\overline{x}_i = cos(\frac{i}{n}\pi), i = 0, 1, ..., n$

For interval [a,b] : $\overline{x}_i = \frac{1}{2}(a+b) + \frac{1}{2}(b-a)cos(\frac{i}{n}\pi), i = 0, 1, ..., n$

One can show that:

where x_k is any other choice of nodes.

Error bound: $|e(x)| \le \frac{1}{(n+1)!}|f^{(n+1)}(x)|2^{-n}$

Example Consider the same example with uniform nodes, $f(x) = sin\pi x$

With Chebyshev nodes, we have

$$|e(x)| \le \frac{1}{(n+1)!}\pi^{n+1}2^{-n}$$

Now look at the much smaller error bounds!

Type II: Chebyshev nodes can be chosen strictly inside the interval (a,b):

n	error bound	measured error
4	1.6×10^{-1}	1.15×10^{-1}
8	3.2×10^{-4}	2.6×10^{-4}
16	1.2×10^{-11}	1.1×10^{-11}

Figure 9:

$$\overline{x}_i = \frac{1}{2}(a+b) + \frac{1}{2}(b-a)cos(\frac{2i+1}{2n+2}\pi), i = 0, 1, ..., n$$

5 Discussion

For large n, polynomials are heavy to deal with.

In general, interpolation polynomials do not coverge as n gets closer to infinity.

For small intervals, the error with polynomial interpolation is small.

Conclusion: Better to use piecewise polynomial interpolation - next chapter (splines). This uses polynomials of not so high power on smaller intervals.