Module 6 – Interpolation

Lesson goals

1. Knowing how to evaluate polynomial coefficients and interpolate with MATLAB's `polyfit` and `polyval` functions.

2. Knowing how to perform an interpolation with Newton's polynomial.

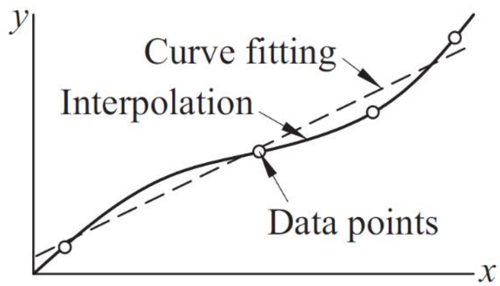
3. Knowing how to perform an interpolation with a Lagrange polynomial.

Introduction

Interpolation vs curve fitting: Discrete data sets, or tables of the form

x_0	x_1	x_2	...	x_n
y_0	y_1	y_2	...	y_n

are commonly involved in technical calculations. The source of the data may be experimental observations or numerical computations. There is a distinction between interpolation and curve fitting. In interpolation we construct a curve through the data points. In doing so, we make the implicit assumption that the data points are accurate and distinct. In contrast, curve fitting is applied to data that contain scatter (noise), usually caused by measurement errors. Here we want to find a smooth curve that approximates the data in some sense. Thus, the curve does not necessarily hit the data points. The difference between interpolation and curve fitting is illustrated in the following Figure



Interpolation

The general formula for an nth-order polynomial can be written as

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

where $a_0, a_1, \ldots, a_n$ are real constants (coefficients) and the exponents are nonnegative integers. For a tabulated function, like the one in the following table, consider the problem of estimating the values of the function at nontabulated points.

x_i	y_i	x_i	y_i
1	1.3	6	8.8
2	3.5	7	10.1
3	4.2	8	12.5
4	5.0	9	13.0
5	6.0	10	15.6

This can be done through a procedure that is called *interpolation*.

Question: Why are the polynomials important? One reason for their importance is that they uniformly approximate continuous functions (Weierstrass Approximation Theorem).

Weierstrass Approximation Theorem: Suppose that f is defined and continuous on $[a, b]$. For each $\varepsilon > 0$, there exists a polynomial $P(x)$, with the property that

$$|f(x) - P(x)| < \varepsilon, \quad \text{for all } x \in [a, b].$$

$(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ at most $\underline{\underline{n}}$

Polynomial interpolation: For $(n+1)$ data points, there is one and only one polynomial of degree at most n (with at most $n+1$ coefficients) that passes through all the points. That is, for $n+1$ data points (x_i, y_i), for $i = 0,1,2,\dots,n$, there is a unique polynomial $P_n(x)$ of degree (at most) n so that

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

$$P_n(x_i) = y_i, \quad i = 0,1,2,\dots,n$$

A straightforward approach for computing the coefficients of this unique polynomial is to solve above system of $(n+1)$ linear equations and $(n+1)$ unknowns for coefficients.

Example. Suppose that we want to determine the coefficients of the parabola,

$$P_2(x) = a_2 x^2 + a_1 x + a_0 \checkmark$$

that passes through the $(300, 0.616)$, $(400, 0.525)$, and $(500, 0.457)$.

Solution. Solving the following system of equations for a_0, a_1, a_2,

$$P_2(300) = 0.616 \rightarrow a_2(300)^2 + a_1(300) + a_0 = 0.616$$

$$P_2(400) = 0.525 \rightarrow a_2(400)^2 + a_1(400) + a_0 = 0.525$$

$$P_2(500) = 0.457 \rightarrow a_2(500)^2 + a_1(500) + a_0 = 0.457$$

leads to $a_0 = 1.027, a_1 = -0.001715, a_2 = 0.00000115$ (do the calculations!).

$$A = \begin{bmatrix} 90000 & 300 & 1 \\ 160000 & 400 & 1 \\ 250000 & 500 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 0.616 \\ 0.525 \\ 0.457 \end{bmatrix}$$

output of MATLAB $\Rightarrow a = \begin{pmatrix} 0 \\ -0.0017 \\ 1.027 \end{pmatrix} \rightarrow P_2(x) = -0.0017x + 1.027$

Remark. The condition number of the above system (use `cond(A)` function in MATLAB) is 5.8932e + 006. This shows that the system is ill-conditioned, so the answer cannot be reliable.

Note. The coefficients of the parabola in the above example can be computed by the MATLAB built-in function `polyfit`. Try the following lines in MATLAB command window:

```
>> T = [300 400 500];
>> density = [0.616 0.525 0.457];
>> p = polyfit(T,density,2)
```

Then, evaluate it at $x = 350$, using the following line:

```
>> d = polyval(p,350)
```

There are alternative approaches that do not manifest this shortcoming.

- Lagrange's method for polynomial interpolation
- Newton's method for polynomial interpolation

Newton's Polynomial Interpolation

$(x_1, f_1), (x_2, f_2)$

Let's get started with simple cases first.

1. **Linear interpolation.** We would like to determine the coefficients of

$$P_1(x) = a_2 x + a_1$$

So that it passes through $(x_1, f(x_1))$ and $(x_2, f(x_2))$. Solve $P_1(x_1) = f(x_1)$ and $P_1(x_2) = f(x_2)$, defines

$$\begin{cases} P(x_1) = f_1 \\ P(x_2) = f_2 \end{cases} \mapsto \begin{cases} a_2 x_1 + a_1 = f_1 \\ a_2 x_2 + a_1 = f_2 \end{cases}$$

$$a_2 = \frac{f(x_2) - f(x_1)}{x_2 - x_1}, \quad a_1 = \frac{x_2 f(x_1) - x_1 f(x_2)}{x_2 - x_1}$$

$$a_2 = \frac{f_2 - f_1}{x_2 - x_1}$$

1^{st} eqn. $\to a_1 = f_1 - a_2 x_1$

$$= f_1 - \frac{f_2 - f_1}{x_2 - x_1} x_1$$

$$= \frac{f_1 x_2 - f_2 x_1}{x_2 - x_1}$$

Thus,

$$P_1(x) = \frac{f(x_2) - f(x_1)}{x_2 - x_1} x + \frac{x_2 f(x_1) - x_1 f(x_2)}{x_2 - x_1}$$

$$= f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1)$$

Note that the coefficient of $(x - x_1)$ in the above rearrangement is the finite difference approximation of the first derivative.

$$P_1(x) = \frac{f_2 - f_1}{x_2 - x_1} x + \frac{x_2 f_1 - f_2 x_1}{x_2 - x_1}$$

$$= \frac{f_2 - f_1}{x_2 - x_1} x + \frac{f_1 x_2 - f_1 x_1 + f_1 x_1 - f_2 x_1}{x_2 - x_1} = f_1 + \frac{f_2 - f_1}{x_2 - x_1}(x - x_1)$$

$= f[x_1, x_2]$

Example. Use $f(x) = \ln x$ in the interval $[1, 6]$ to estimate $\ln 2$ using linear interpolation.

x_i	x_1: 1	x_2: 6
$f(x_i)$	0 (f_1)	1.7918 (f_2)

$$P_1(x) = f_1 + \frac{f_2 - f_1}{x_2 - x_1}(x - x_1)$$

$$= 0 + \frac{1.7918 - 0}{6 - 1}(x - 1) = 0.3584(x-1)$$

$$\ln(2) \simeq P_2(2) = 0.3584(2-1) = 0.3584$$

2. **Quadratic interpolation.** We are looking for a particular quadratic polynomial that passes through the points $(x_1, f(x_1))$, $(x_2, f(x_2))$, and $(x_3, f(x_3))$ and has the form as below:

(x_1, f_1)
(x_2, f_2)
(x_3, f_3)

$$P_2(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2)$$

Simple procedure:

$P(x_1) = f_1 \to b_1 = f_1$

$P(x_2) = f_2 \to b_1 + b_2(x_2 - x_1) = f_2$

$P(x_3) = f_3 \Rightarrow b_1 + b_2(x_3 - x_1) + b_3(x_3 - x_1)(x_3 - x_2) = f_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & x_2 - x_1 & 0 \\ 1 & x_3 - x_1 & (x_3 - x_2)(x_3 - x_1) \end{bmatrix}$$

- $x = x_1$ leads to $b_1 = f(x_1)$
- $x = x_2$ and $b_1 = f(x_1)$ lead to
$$\boxed{b_2 = \frac{f(x_2) - f(x_1)}{x_2 - x_1} := f[x_2, x_1]}$$
- $x = x_3$, $b_1 = f(x_1)$ and the formula for b_2 lead to
$$b_3 = \frac{\frac{f(x_3) - f(x_2)}{x_3 - x_2} - \frac{f(x_2) - f(x_1)}{x_2 - x_1}}{x_3 - x_1} = \frac{f[x_3, x_2] - f[x_2, x_1]}{x_3 - x_1} := f[x_3, x_2, x_1]$$

$$b_3 = \frac{f_3 - b_1 - b_2(x_3 - x_1)}{(x_3 - x_1)(x_3 - x_2)} = \frac{f_3 - f_1 - \frac{f_2 - f_1}{x_2 - x_1}(x_3 - x_1)}{(x_3 - x_1)(x_3 - x_2)} = \frac{\frac{f_3 - f_1}{x_3 - x_2} - \frac{f_2 - f_1}{(x_2 - x_1)(x_3 - x_2)}(x_3 - x_1)}{(x_3 - x_1)}$$

$$= \frac{\frac{f_3-f_1}{x_3-x_2} - \frac{f_2-f_1}{(x_2-x_1)(x_3-x_2)}\left(x_3-x_2+x_2-x_1\right)}{x_3-x_1} = \frac{\frac{f_3-f_1}{x_3-x_2} - \frac{f_2-f_1}{x_2-x_1} - \frac{f_2-f_1}{x_3-x_2}}{x_3-x_1}$$

$$= \frac{\frac{f_3-f_2}{x_3-x_2} - \frac{f_2-f_1}{x_2-x_1}}{x_3-x_1}$$

Example. Use $f(x) = \ln x$ at the points $x = 1, 4, 6$ to estimate $\ln 2$ using quadratic interpolation.

x_i	1	4	6
f_i	0	1.3863	1.7918

$$P_2(x) = b_1 + b_2(x-1) + b_3(x-1)(x-4)$$

$b_1 = f_1 = 0$

$b_2 = f[x_2, x_1] = \frac{f_2-f_1}{x_2-x_1} = 0.4621$, $f[x_3, x_2] = \frac{f_3-f_2}{x_3-x_2} = 0.2028$

$$b_3 = \frac{f[x_3,x_2] - f[x_2,x_1]}{x_3-x_1} = \frac{0.2028 - 0.4621}{5} = -0.0519$$

$$P_2(x) = 0.4621(x-1) - 0.0519(x-1)(x-4)$$

$$\ln(2) \simeq P_2(2) = 0.4621 - (0.0519)(-2) = 0.5659$$

$(x_1, f_1), (x_2, f_2), \ldots, (x_4, f_n)$

n points

3. General form of Newton's method. The preceding analysis can be generalized to fit an $(n-1)$th degree polynomial to n data points. The $(n-1)$th degree polynomial is

$$P_{n-1}(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2) + \cdots + b_n(x - x_1)(x - x_2)\ldots(x - x_{n-1})$$

Let

$$b_1 = f[x_1]$$
$$b_2 = f[x_2, x_1] = \frac{f_2 - f_1}{x_2 - x_1}$$
$$b_3 = f[x_3, x_2, x_1] = \frac{f[x_3, x_2] - f[x_2, x_1]}{x_3 - x_1}$$
$$\vdots$$
$$b_n = f[x_n, x_{n-1}, \ldots, x_2, x_1]$$

where the bracketed function evaluations are called finite divided differences and are computed as below:

$$f[x_i, x_j] = \frac{f(x_i) - f(x_j)}{x_i - x_j} \qquad \text{first divided difference}$$

$$f[x_i, x_j, x_k] = \frac{f[x_i, x_j] - f[x_j, x_k]}{x_i - x_k} \qquad \text{second divided difference}$$

$$\vdots$$

$$f[x_n, x_{n-1}, \ldots, x_2, x_1] = \frac{f[x_n, x_{n-1}, \ldots, x_2] - f[x_{n-1}, \ldots, x_2, x_1]}{x_n - x_1} \qquad \text{(n − 1)th divided difference}$$

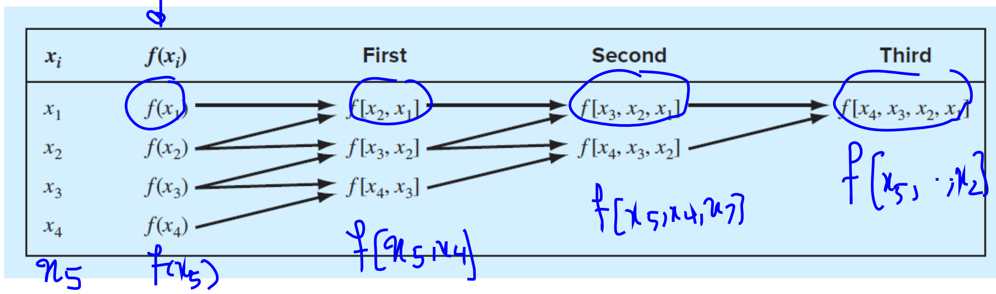
These differences can be used to evaluate the coefficients to yield the general form of Newton's interpolating polynomial:

$$P_{n-1}(x) = f[x_1] + f[x_2, x_1](x - x_1) + \cdots + f[x_n, \ldots, x_2, x_1](x - x_1)(x - x_2)\ldots(x - x_{n-1})$$

Note that it is not necessary that the data points be equally spaced. Also, using above recursive formulas, higher-order differences are computed by taking differences of lower-order differences.

Zero divided diff.

Fourth $f[x_5, \ldots, x_1]$



Graphical depiction of the recursive nature of finite divided differences. This representation is referred to as a divided difference table.

Example. For $f(x) = \ln x$, we used the points $x = 1, 4, 6$ to estimate $\ln 2$ using quadratic interpolation. Now, add one more point at $x = 5$ to estimate $\ln 2$ using third-order Newton's method.

Solution.

x_i	$f(x_i)$	First	Second	Third
1	0	0.4620981	−0.05187311	0.007865529
4	1.386294	0.2027326	−0.02041100	
6	1.791759	0.1823216		
5	1.609438			

new data

For 3 points $(1, 0), (4, 1.3863), (6, 1.7918)$

x_i	f_i	$f[x_i, x_j]$	$f[x_i, x_j, x_k]$
1	0	$\frac{1.3863 - 0}{4-1} = 0.4621$	$\frac{0.2027 - 0.4621}{6-1} = -0.0519$
4	1.3863	$\frac{1.7918 - 1.3863}{6-4} = 0.2027$	
6	1.7918		

$$P_3(x) = 0 + 0.4621(x-1) - 0.0519(x-1)(x-4) + 0.0079(x-1)(x-4)(x-6)$$

Lagrange's Polynomial Interpolation

$(x_1, f_1), (x_2, f_2)$

- **Linear Lagrange interpolating polynomial.** Suppose we formulate a linear interpolating polynomial as the weighted average of the two values that we are connecting by a straight line:

$P_1(x) = L_1(x) f_1 + L_2(x) f_2$

$$P_1(x) = L_1(x)f(x_1) + L_2(x)f(x_2)$$

It is logical that the first weighting coefficient is the straight line that is equal to 1 at x_1 and 0 at x_2. So,

x_1, x_2

$$L_1(x) = \frac{x - x_2}{x_1 - x_2}$$

$L_1(x) = \frac{x-x_2}{x_1-x_2}$

Similarly,

$$L_2(x) = \frac{x - x_1}{x_2 - x_1}$$

$L_2(x) = \frac{x-x_1}{x_2-x_1}$

Thus,

$$P_1(x) = \frac{x - x_2}{x_1 - x_2} f(x_1) + \frac{x - x_1}{x_2 - x_1} f(x_2)$$

$(x-x_1)(x-x_2)(x-x_3)$

$(x_1, f_1), (x_2, f_2), (x_3, f_3)$

x_1, x_2, x_3

- **Quadratic Lagrange interpolating polynomial.** The same strategy can be employed to fit a parabola through three points. If

$L_1(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)}$

$$P_2(x) = L_1(x)f(x_1) + L_2(x)f(x_2) + L_3(x)f(x_3)$$

Then, we have

$$L_1(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}, \quad L_2(x) = \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}, L_3(x) = \frac{(x - x_2)(x - x_1)}{(x_3 - x_2)(x_3 - x_1)}$$

x_1, x_2, x_3, x_4

$L_1(x) = \frac{(x-x_2)(x-x_3)(x-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)}$

$(x_1, f_1), \ldots (x_n, f_n)$

- **General form of Lagrange interpolating polynomial.** In general, the $(n-1)$th degree Lagrange interpolating polynomial that passes through the n points is given by:

$$P_{n-1}(x) = \sum_{i=1}^{n} L_i(x) f(x_i)$$

where, for $i = 1,2, \ldots, n$, the $(n-1)$th degree Lagrange's polynomial $L_i(x)$ is defined as below:

$$L_i(x) = \frac{(x - x_1) \ldots (x - x_{i-1})(x - x_{i+1}) \ldots (x - x_n)}{(x_i - x_1) \ldots (x_i - x_{i-1})(x_i - x_{i+1}) \ldots (x_i - x_n)} = \prod_{j=1, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

Example. Use a Lagrange interpolating polynomial of the first and second order to evaluate the density of unused motor oil at $T = 15$ °C based on the following data:

$$(0, 3.85), \qquad (20, 0.800), \qquad (40, 0.212)$$

$(x-x_1)(x-x_2)(x-x_3)$

$$P_2(x) = f_1 L_1(x) + f_2 L_2(x) + f_3 L_3(x)$$

$$= (3.85) L_1(x) + (0.8) L_2(x) + (0.212) L_3(x)$$

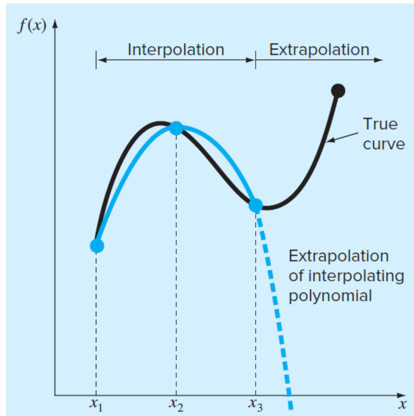
$$= (3.85) \frac{(x-20)(x-40)}{(0-20)(0-40)} + (0.8) \frac{(x-0)(x-40)}{(20-0)(20-40)} + (0.212) \frac{(x-0)(x-20)}{(40-0)(40-20)}$$

$$= \frac{3.85}{800}(x-20)(x-40) - \frac{0.8}{400}(x)(x-40) + \frac{0.212}{800}(x)(x-20)$$

$$f(15) \simeq P_2(15) = \frac{3.85}{800}(5)(25) - \frac{0.8}{400}(15)(-25) + \frac{0.212}{800}(15)(-5) = 1.332$$

Issues with polynomial interpolation.

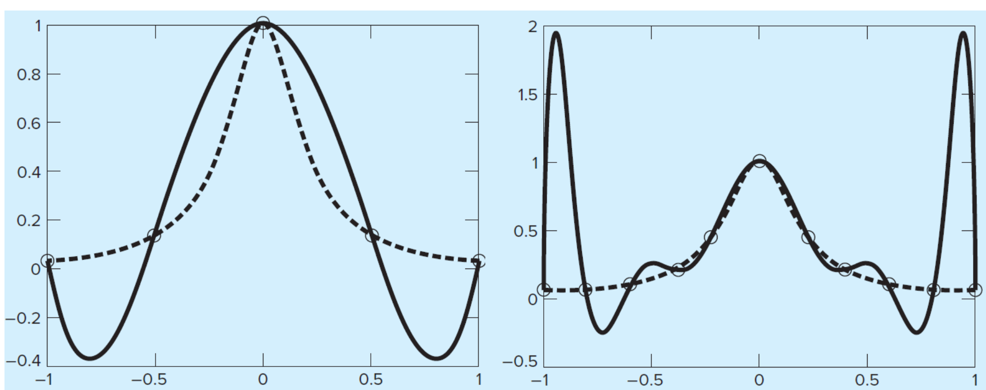
Extrapolation. It is the process of estimating a value of $f(x)$ that lies outside the range of the known base points, $x_1, x_2, \dots, x_n$.



Extreme care should be exercised in extrapolation as the true curve could easily diverge from the prediction.

Oscillations. Although “more is better” in many contexts, it is absolutely not true for polynomial interpolation. Higher-order polynomials tend to be very ill-conditioned—that is, they tend to be highly sensitive to round off error. The following example shows the oscillating behavior of the polynomial interpolation for the nice looking function

$$f(x) = \frac{1}{1 + 25x^2}$$



Although there may be certain contexts where higher-order polynomials are necessary, they are usually to be avoided. In most engineering and scientific contexts, lower order polynomials can be used effectively to capture the curving trends of data without suffering from oscillations.

References

1. Chapra, Steven C. (2018). *Numerical Methods with* MATLAB *for Engineers and Scientists*, 4th Ed. McGraw Hill.
2. Burden, Richard L., Faires, J. Douglas (2011). *Numerical Analysis*, 9th Ed. Brooks/Cole Cengage Learning