Interpolation using the Vandermonde matrix

The most basic procedure to determine the coefficients $a_0, a_1, \ldots, a_n$ of a polynomial

$$P_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

such that it interpolates the $n+1$ points

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$

is to write a linear system of equations as follows:

$$\begin{aligned}
P_n(x_0) = y_0 \quad &\Rightarrow \quad a_o + a_1 x_0 + a_2 x_0^2 + \cdots + a_{n-1} x_0^{n-1} + a_n x_0^n = y_0 \\
P_n(x_1) = y_1 \quad &\Rightarrow \quad a_o + a_1 x_1 + a_2 x_1^2 + \cdots + a_{n-1} x_1^{n-1} + a_n x_1^n = y_1 \\
\vdots \quad &\Rightarrow \quad \vdots \\
P_n(x_n) = y_n \quad &\Rightarrow \quad a_o + a_1 x_n + a_2 x_n^2 + \cdots + a_{n-1} x_n^{n-1} + a_n x_n^n = y_n
\end{aligned}$$

or, in matrix form:

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^{n-1} & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-1} & x_{n-1}^n \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-1} & x_n^n \end{bmatrix}}_{\mathbf{V}} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix}}_{\vec{a}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}}_{\vec{b}}$$

The matrix $\mathbf{V}$ is called a *Vandermonde matrix*. We sill see that $\mathbf{V}$ is non-singular, thus we can solve the system $\mathbf{V}\vec{a} = \vec{y}$ to obtain the coefficients $\vec{a} = (a_0, a_1, \ldots, a_n)$. Let's evaluate the merits and drawbacks of this approach:

- Cost to determine the polynomial $P_n(x)$: *VERY COSTLY* since a dense $(n+1) \times (n+1)$ linear system has to be solved. This will generally require time proportional to n^3, making large interpolation problems intractable. In addition, the Vandermonde matrix is notorious for being challenging to solve (especially with Gauss elimination) and prone to large errors in the computed coefficients a_i when n is large and/or $x_i \approx x_j$.

- Cost to evaluate $f(x)$ ($x =$ arbitrary) if coefficients are known: *VERY CHEAP*. Using Horner's scheme:

$$a_0 + a_1 x + \cdots + a_n x^n = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + x a_n)))$$

- Availability of derivatives: *VERY EASY*, e.g.

$$P_n'(x) = a_1 + 2a_2 x + 3a_3 x^2 + \cdots + (n-1)a_{n-1} x^{n-2} + n a_n x^{n-1}$$

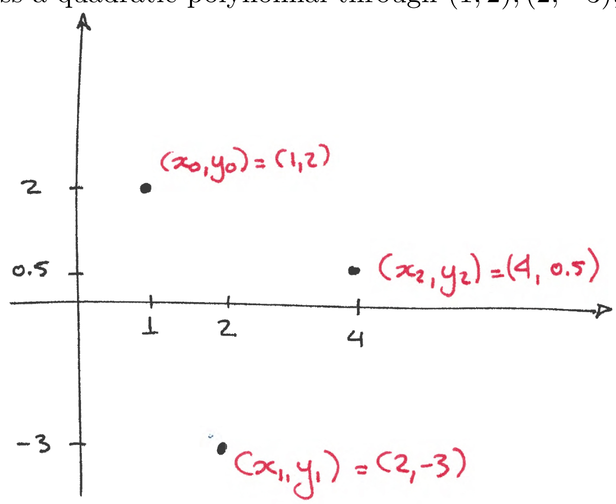
- Support for incremental interpolation: *NOT SUPPORTED!* This property examines if interpolating through $(x_1, y_1), \ldots, (x_{n+1}, y_{n+1})$ is *easier* if we already know a polynomial (of degree $= n-1$) that interpolates through $(x_1, y_1), \ldots, (x_n, y_n)$. In our case, the system $\mathbf{V}\vec{a} = \vec{y}$ would have to be solved from scratch for the $(n+1)$ data points.

Lagrange interpolation

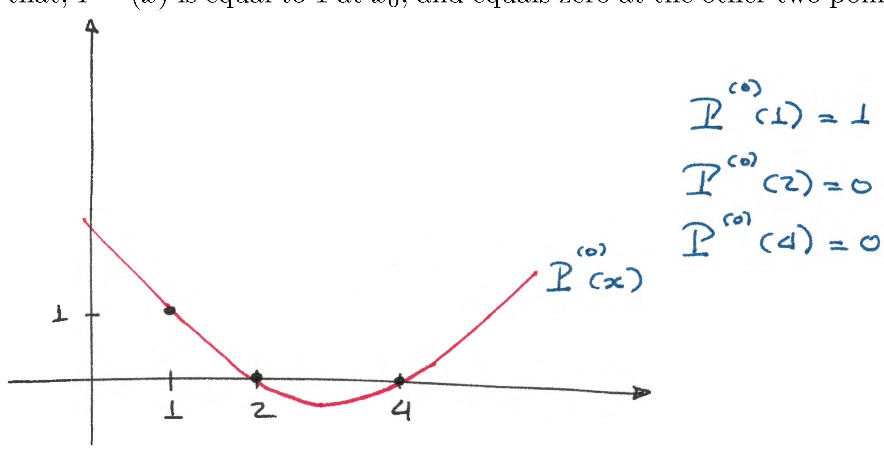
Corresponding textbook chapter(s): §4.3

The Lagrange interpolation method is an alternative way to define $P_n(x)$ without having to solve computationally expensive systems of equations. We shall explain how Lagrange interpolation works with an example.

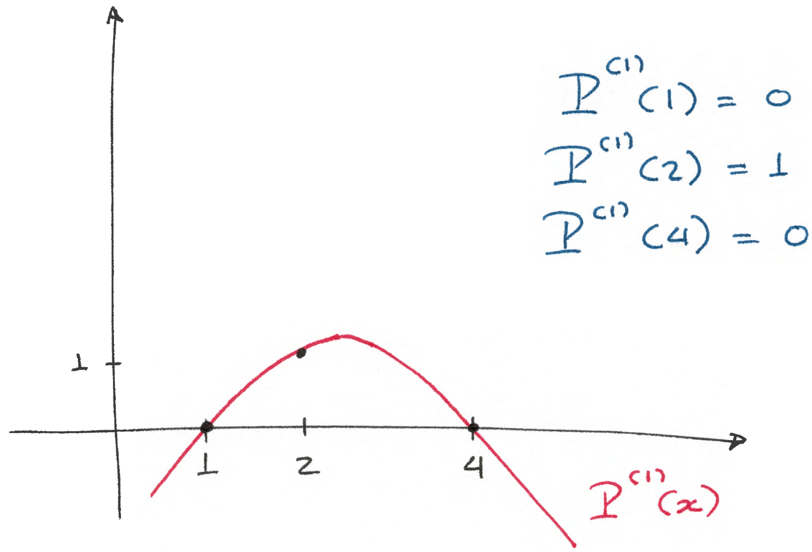
Example: Pass a quadratic polynomial through $(1, 2), (2, -3), (4, 0.5)$.



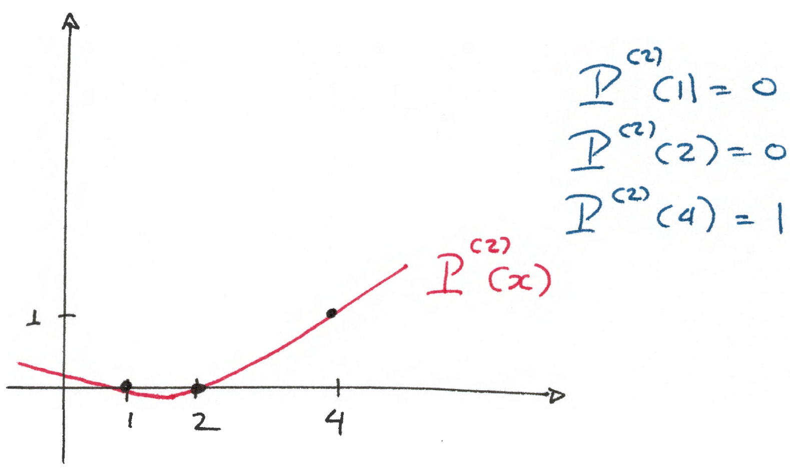
Assume we have somehow constructed 3 quadratic polynomials $P^{(0)}(x), P^{(1)}(x), P^{(1)}(x)$, such that, $P^{(0)}(x)$ is equal to 1 at x_0, and equals zero at the other two points x_1, x_2:



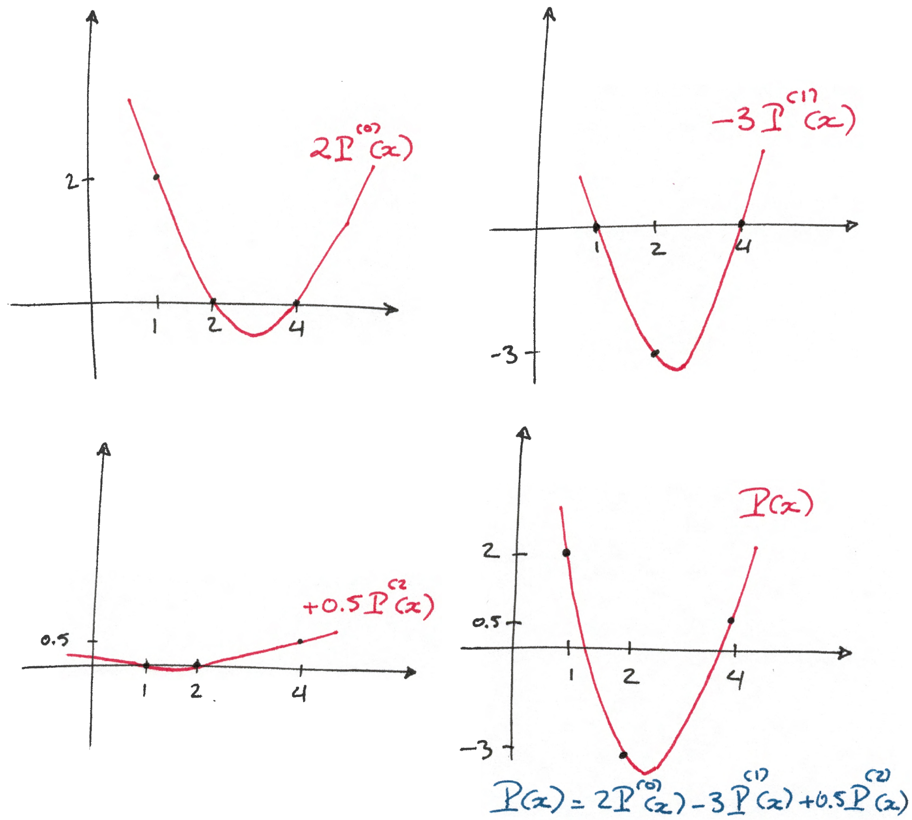
$P^{(1)}(x)$ is designed as to equal 1 at location x_1, and evaluate to zero at x_0, x_2:



While $P^{(2)}$ is similarly constructed to satisfy



Now, the idea is to *scale* each $P^{(i)}$, such that $P^{(i)}(x_i) = y_i$ and add them all together:



In summary, if we have a total of $(n+1)$ data points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, define the Lagrange polynomials of n-degree $l_0(x), l_1(x), \ldots, l_n(x)$ as:

$$l_i(x_j) = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{cases} \tag{6}$$

Then, the interpolating polynomial is simply:

$$P(x) = y_0 l_0(x) + y_1 l_1(x) + \cdots + y_n l_n(x) = \sum_{i=0}^{n} y_i l_i(x).$$

No solution of a linear system is necessary here. We just have to explain what every $l_i(x)$ looks like. Since $l_i(x)$ is an n-degree polynomial with n roots

$$x_0, x_1, x_2, \ldots, x_{i-1}, x_{i+1}, x_{i+2}, \ldots, x_n,$$

it must have the form

$$\begin{aligned} l_i(x) &= C_i(x - x_0)(x - x_1)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n) \\ &= C_i \prod_{j \neq i}(x - x_j) \end{aligned}$$

Now, we require $l_i(x_k) = 1$, thus:

$$1 = C_i \prod_{j \neq i} (x_i - x_j) \Rightarrow C_i = \frac{1}{\prod_{j \neq i} (x_i - x_j)}.$$

Thus, for every i, we have:

$$\begin{aligned} l_i(x) &= \frac{(x - x_o)(x - x_1) \ldots (x - x_{i-1})(x - x_{i+1})/ldots(x - x_n)}{(x_i - x_o)(x_i - x_1) \ldots (x_i - x_{i-1})(x_i - x_{i+1})/ldots(x_i - x_n)} \\ &= \prod_{j \neq i} (\frac{x - x_j}{x_i - x_j}) \\ &= \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)} \end{aligned}$$

Note: This result essentially proves *existence* of a polynomial interpolant of degree $= n$ that passes through $(n + 1)$ data points. We can also use it to prove that the Vandermonde matrix V is non-singular; if it *were* singular, a right-hand-side $\vec{y} = (y_0, \ldots, y_n)$ would have existed such that $V\vec{a} = \vec{y}$ would have no solution, which is a contradiction.

Let's evaluate the same 4 quality metrics we saw before for the Vandermonde matrix approach.

- Cost of determining $P(x)$: *VERY EASY*. We are essentially able to write a formula for $P(x)$ without solving any systems. However, if we want to write $P(x) - a_0 + a_1 s + \cdots + a_n x^n$, the cost of evaluating the a_i's would be very high! Each l_i would need to be expanded $\Rightarrow$ approximately N^2 operations for each l_i, N^3 operations for $P(x)$.

- Cost of evaluating $P(x)$ (x = arbitrary): *SIGNIFICANT*. We do not really need to compute the a_i's beforehand if we only need to evaluate $P(x)$ at select few locations. For each $l_i(x)$ the evaluation requires N subtractions and N multiplications $\Rightarrow$ total = about N^2 operations (better than N^3 for computing the a_i's).

- Availability of derivatives: *NOT READILY AVAILABLE*. Differentiating each l_i (since $P'(x) = \sum y_i l_i'(x)$) is not trivial $\Rightarrow$ yeilds N terms each with $(N - 1)$ products per term.

- Incremental interpolation: The Lagrange method does not provide any special shortcuts to adding one extra point to the interpolation problem, however it is very easy to simply rebuild the new interpolant $P(x)$ from scratch.