Least Squares

Approximation by the least square method

Suppose we have $n+1$ points $x_0, x_1, \ldots, x_n$ and $n+1$ values $y_0, y_1, \ldots, y_n$. We have seen that if n is large, the interpolating polynomial may show large oscillations.

One solution could be to break the interpolation domain in pieces and then perform a multiple interpolation. Or, instead of interpolating the values, it is possible to define a polynomial of degree $m < n$ that approximates the data "at best".

Definition 1. *We call **least squares polynomial approximation of degree** m the polynomial $\tilde{f}_m(x)$ of degree m such that*

$$\boxed{\sum_{i=0}^{n} |y_i - \tilde{f}_m(x_i)|^2 \leq \sum_{i=0}^{n} |y_i - p_m(x_i)|^2} \quad \forall p_m(x) \in \mathbb{P}^m$$

Remark 1. *If $y_i = f(x_i)$ with f a continuous function, then $\tilde{f}_m$ is called the **approximation of f in the least squares sense**, or **least squares approximation of f**.*

In other words, the least squares polynomial approximation is the polynomial of degree m that minimizes the distance from the data points.
Let note the polynomial $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$ with the $m+1$ coefficients a_i unknown, and define the **loss function** as

$$\Phi(a_0, a_1, \ldots, a_m) = \sum_{i=0}^{n} |y_i - \tilde{f}_m(x_i)|^2 = \sum_{i=0}^{n} |y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_m x_i^m)|^2$$

Since we want to minimize the loss function Φ, we put the derivative w.r.t. the coefficients to zero. Then the coefficients of $\tilde{f}_m$ can be determined by the relation

$$\frac{\partial \Phi}{\partial a_k} = 0, \quad k = 0, \ldots, m, \tag{1}$$

i.e., $m+1$ linear equations with $m+1$ unknowns a_k, $k = 0, \ldots, m$, which means that the problem admits an unique solution, so it is well posed.

Ideally, for $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$ we would like to impose $\tilde{f}_m(x_i) = y_i$ for $i = 0, \ldots, n$. This can be written as a linear system with basis $1, x, x^2, \ldots, x^m$ and unknowns $a_k, k = 0, \ldots, m$: $B\mathbf{a} = y$, where B is a matrix of dimension $(n+1) \times (m+1)$, called **Vandermonde matrix**:

$$B = \begin{pmatrix} 1 & x_0 & \ldots & x_0^m \\ 1 & x_1 & \ldots & x_1^m \\ \vdots & & & \vdots \\ 1 & x_n & \ldots & x_n^m \end{pmatrix}$$

Since $m < n$, the system is oversized. The solution to (1) is equivalent to the square system (**system of normal equations**)

$$\boxed{B^T B\mathbf{a} = B^T y}$$

While $\tilde{f}_m$ is a polynomial, we can generalize the formula for functions of a space V_m obtained by linearly combining $m+1$ independent functions $\{\psi_j, j=0,1,\ldots,m\}$. The choice of ψ is dictated by the conjectured behaviour of the function underlying the current data distribution. So we have that $\tilde{f}(x) = \sum_{j=0}^{m} a_j \psi_j(x)$ and the unknown coefficients $a = (a_0, a_1, \ldots, a_m)$ can be obtaining solving the system $B^T B a = B^T y$ where in this case $B = b_{ij} = \psi_j(x_i)$ and y are the data.

Generalization

We would like to approximate a function evaluated on a (large) number of data points, using a finite dimensional space V_h of dimension n, defined as the *span* of a set of basis functions v_i: any function in V_h can be expressed as a linear combination of the basis v_i:

$$v_h(x) = v^i v_i(x)$$

where summation is implied on i (Einstein notation).

Assume we'd like to approximate the function $f : \Omega \mapsto \mathbb{R}$ and that the only thing we have at our disposal is N pairs (x_i, y_i), i.e., N points $x_i \in \Omega$ in which we know the values $f(x_i) = y_i$.

Given *any* finite dimensional space V_h of dimension n (i.e., any collection of n *linearly independent* functions $v_i : \Omega \mapsto \mathbb{R}$), we define the **basis collocation matrix** B as the rectangular matrix

$$B_{ij} = v_j(x_i), \quad i = 1, \ldots, N, \; j = 1, \ldots, n.$$

An element of V_h evaluated in all points x_i can be computed easily by the matrix vector product between B and the vector of coefficients v:

$$v_h(x_i) = (Bv)_i = B_{ij} v^j = v^j v_j(x_i)$$

Computing the **least square approximation** of f in V_h is equivalent to finding the element of V_h that minimizes the following functional:

$$E(v_h) := \frac{1}{2N} \sum_{i=1}^{N} |v_h(x_i) - y_i|^2 \tag{2}$$

where $E(v_h)$ is the **mean squared error (MSE)** or **mean squared deviation (MSD)** of the approximation v_h, i.e., **the average of the squares of the errors**—that is, the **average squared difference between the approximated values and the actual value**.

Expressing $v_h(x_i)$ with the matrix product, $E(v_h)$ can be written as

$$E(v_h) := \frac{1}{2N} (Bv - y)^T (Bv - y) \tag{3}$$

If we want to minimize E, we can take its derivative w.r.t. the coefficients v^i and set it to zero, i.e.:

$$\frac{\partial E}{\partial v^i} = \frac{1}{2N} \frac{\partial [(Bv - y)^T (Bv - y)]}{\partial v^i} = \frac{1}{N} (B^T B v - B^T y) = 0$$

which admits a unique solution if the following linear system has a solution:

$$B^T B v = B^T y. \tag{5}$$