The Least Squares Approximation Method

(Reading: Cheney & Kincaid, Sections 9.1-9.2)

Motivation: Suppose we are given $m+1$ data points: (x_0, y_0), (x_1, y_1),..., (x_m, y_m). Often y_i is a measurement from some sensors or surveys at x_i for $i = 0, 1, \dots, m$. These measurements have errors. It doesn't make much sense to request an approximate function to pass through the data points $(x_0, y_0), (x_1, y_1), \dots, (x_m, y_m)$ exactly. We would like to find an approximate function such that all points are as "close" to the function as possible.

Data Fitting by a Straight Line

Suppose the data are thought to conform to a linear relationship:

assumption
$$y = ax + b,$$

where a and b are two unknown parameters to be determined. We use this straight line to fit the data. One of the typical approaches to requiring all points be as "close" to the line as possible is to solve the following optimization problem

$$\min_{a,b} \phi(a, b), \qquad \phi(a, b) \equiv \sum_{k=0}^{m} (ax_k + b - y_k)^2.$$

optimal

This optimization problem is called a *least squares* problem.

From calculus, the conditions that

$$\frac{\partial \phi}{\partial a} = 0, \qquad \frac{\partial \phi}{\partial b} = 0$$

are necessary at the minimum. Then from the expression of $\phi(a, b)$, we can easily obtain

$$\begin{bmatrix} \sum_{k=0}^{m} x_k^2 & \sum_{k=0}^{m} x_k \\ \sum_{k=0}^{m} x_k & m+1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{m} x_k y_k \\ \sum_{k=0}^{m} y_k \end{bmatrix}.$$

The above equations are called the *normal equations*, which can be easily solved. In fact, it is easy to show

$$\begin{aligned} a &= \left[(m+1) \sum_{k=0}^{m} x_k y_k - (\sum_{k=0}^{m} x_k)(\sum_{k=0}^{m} y_k) \right] \Big/ \left[(m+1) \sum_{k=0}^{m} x_k^2 - (\sum_{k=0}^{m} x_k)^2 \right] \\ b &= \left[(\sum_{k=0}^{m} x_k^2)(\sum_{k=0}^{m} y_k) - (\sum_{k=0}^{m} x_k)(\sum_{k=0}^{m} x_k y_k) \right] \Big/ \left[(m+1) \sum_{k=0}^{m} x_k^2 - (\sum_{k=0}^{m} x_k)^2 \right]. \end{aligned}$$

Data Fitting by a General Linear Family of Functions

Suppose the data are thought to conform to a relationship like

assumption
$$y = \sum_{j=0}^{n} c_j g_j(x),$$

$1, x, x^2 \ldots$

where the functions $g_0, g_1, \ldots, g_n$ (called basis functions) are known, the coefficients $c_0, c_1, \ldots, c_n$ are to be determined, and $m \geq n$. To determine $c_0, c_1, \ldots, c_n$, we solve the least squares problem

$$\min_{c_0, c_1, \ldots, c_n} \phi(c_0, c_1, \ldots, c_n), \qquad \phi(c_0, c_1, \ldots, c_n) \equiv \sum_{k=0}^{m} \Big(\sum_{j=0}^{n} c_j g_j(x_k) - y_k\Big)^2. \tag{1}$$

From the expression of ϕ, we obtain

$$\frac{\partial \phi}{\partial c_i} = \sum_{k=0}^{m} 2\Big[\sum_{j=0}^{n} c_j g_j(x_k) - y_k\Big] g_i(x_k), \qquad i = 0, 1, \ldots, n.$$

In order to find the optimal solution, we set $\frac{\partial \phi}{\partial c_i} = 0$ for $i = 0, 1, \ldots, n$, leading to the normal equations:

$$\sum_{j=0}^{n} \Big[\sum_{k=0}^{m} g_i(x_k) g_j(x_k)\Big] c_j = \sum_{k=0}^{m} g_i(x_k) y_k, \qquad i = 0, 1, \ldots, n. \tag{2}$$

Here we have a linear system of equations. In principle, (2) can be solved by Gaussian elimination with no pivoting. But better algorithms (which are taught in COMP 540 and COMP 642) are available. In the following we will show how to compute the solution by MATLAB. In order to do it, we will rewrite the least squares problem and the normal equations by using the matrix-vector language.

Define

matrix form

$$A = \begin{bmatrix} g_0(x_0) & g_1(x_0) & \cdots & g_n(x_0) \\ g_0(x_1) & g_1(x_1) & \cdots & g_n(x_1) \\ \vdots & \vdots & \cdots & \vdots \\ g_0(x_m) & g_1(x_m) & \cdots & g_n(x_m) \end{bmatrix}, \quad c = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}, \quad y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{bmatrix}.$$

Then it is easy to see that the least squares problem (1) can also be written as

$$\min_{c} \|Ac - y\|_2^2.$$

and the normal equations (2) can be written as

$$A^T A c = A^T y. \tag{3}$$

It can be shown that when the columns of A are linearly independent (this is a reasonable assumption), (3) has a unique solution.

In MATLAB, we can compute the least squares solution simply by the command `c = A\y`

Remark: A special case of the least squares fitting problem is $g_j(x) = x^j$ for $j = 0, 1, \ldots, n$. If $n = m$, then the problem is just polynomial interpolation.