Linear regression for function approximation : a crash course

SysID Project
Part 1
Background

We want to find an approximator $\hat{g}(x)$ for an unknown function $g(x)$, from a set of input-output samples:

$$\{(x_i, y_i = g(x_i) + w_i) \mid i = 1, \dots, N\}$$

where $g(x_i)$ is corrupted by noise w_i.

Notations:

$\varphi^T(x) \in \mathbb{R}^n$ $\quad \theta \in \mathbb{R}^n$

We choose a linearly parameterized form for $\hat{g}(x)$:

$$\hat{g}(x) = \varphi_1(x)\theta_1 + \varphi_2(x)\theta_2 + \dots + \varphi_n(x)\theta_n = [\varphi_1(x) \dots \varphi_n(x)] \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$$

where $\varphi_i(x)$ are the regressors (we choose them manually) and θ_i the parameters (the algorithm will find them automatically).

We write our objective : $\hat{g}$ should match y_i for the samples we have.

$$\hat{g}(x_1) = \varphi_1(x_1)\theta_1 + \dots + \varphi_n(x_1)\theta_n = y_1$$
$$\hat{g}(x_2) = \varphi_1(x_2)\theta_1 + \dots + \varphi_n(x_2)\theta_n = y_2$$
$$\vdots$$
$$\hat{g}(x_N) = \varphi_1(x_N)\theta_1 + \dots + \varphi_n(x_N)\theta_n = y_N$$

or, in matrix form:

$$\underbrace{\begin{bmatrix} \varphi_1(x_1) & \dots & \varphi_n(x_1) \\ \varphi_1(x_2) & \dots & \varphi_n(x_2) \\ \vdots & & \\ \varphi_1(x_N) & \dots & \varphi_n(x_N) \end{bmatrix}}_{\Phi \in \mathbb{R}^{N \times n}} \cdot \underbrace{\begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}}_{\theta \in \mathbb{R}^n} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{Y \in \mathbb{R}^N}$$

Notations:

This system is overdetermined, $N > n$, so we cannot solve it with equality.
We will minimize the mean of squared errors between the two sides:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \left[\hat{g}(x_i) - y_i\right]^2$$

In Matlab, it is easy : the parameter vector θ minimizing the MSE is found by writing $\theta = \Phi \backslash Y$ (back slash).

Once θ is found like this, then $\hat{g}$ can be applied to approximate g at any point x_q (not just those in the dataset). Just compute:

$$y_q = \hat{g}(x_q) = \sum_{i=1}^{n} \varphi_i(x_q)\theta_i = \varphi^T(x_q)\theta \quad (T \text{ stands for "query"})$$

To be sure that the solution is good, we'll generally compute the MSE on a different, validation dataset:

$$\{(x_i^V, y_i^V) \mid i = 1, \dots, N^V\}$$

(and plot $\hat{g}$ versus g visually).