EECS 16A Designing Information Devices and Systems I
Spring 2017 Official Lecture Notes Note 17

Introduction

In our previous exploration of linear algebra, we used techniques like Row Reduction and Matrix Inversion to solve explicit systems of equations in multiple variables. However, what happens when we introduce "noise", or imprecision into our system. How do we solve our system of equations in such a way that we minimize error, and get as precise a solution as we possibly can?

In this section we will explore **Least Squares**, a method through which we can minimize error by solving overdetermined systems of equations (more equations than variables). We will also learn how to find the **Minimum Norm Solution** of an underdetermined system of equations (fewer equations than variables).

Key Concepts

By the end of this note, you should be able to do the following:

1. Understand why we would need to use Least Squares, especially in the context of our GPS Locationing systems.
2. Explain how the Least Squares method minimizes the error term of a system of equations.

Least Squares

Let's think back to our GPS system from Lecture, to develop some intuition as to why we actually need a system like Least Squares: Given some number of beacons and the (possibly erroneous) readings of distances between a user and the 3 beacons, how is it possible to determine which distance reading is correct? It turns out that if we add more beacons to get more data, we can make more robust measurements and use the least squares approach to find out the answer.

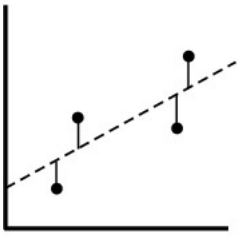
Another example: Suppose you have n points $(a_1, b_1), \cdots, (a_n, b_n)$ and want to find fit a straight line to these points. If all these points lie on the line, then they will all satisfy the equation $b_i = x_1 a_i + x_2$ for some unknown x_1 and x_2. Written in matrix form:

$$\begin{bmatrix} a_1 & 1 \\ a_2 & 1 \\ \vdots & \vdots \\ a_n & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

However not all points may lie on the same straight line. In this case we have n equations and 2 unknowns. If $n > 2$, the system of equations may have no solutions. A general principle to apply when there's more equations than unknowns is to introduce more unknowns so that the equations can be solved. In this case, we introduce an unknown "fudge factor", e_i, for each equation, so that the equations become:

$$b_i + e_i = x_1 a_i + x_2$$

This corresponds to the vertical distance between the best-fit line and points in the following diagram:



We then try to find the line that minimizes the squared vertical distance between each point and the line.

Derivation

Let's think about how we would setup a system in which we have more equations (n) than unknowns (m).

$$a_{11}x_1 + a_{12}x_2 + ... + a_{1m}x_m = b_1 + e_1 \tag{1}$$
$$a_{21}x_1 + ... + ... + a_{2m}x_m = b_2 + e_2 \tag{2}$$
$$a_{n1}x_1 + ... + ... + a_{nm}x_m = b_n + e_n \tag{3}$$

With the above equations, we are trying to approximate a solution to the system with a range for the error. Essentially, we know b (in the above example it would be the distance readings), but we already know that it's going to be a little bit off, so we capture the error with the e term.

We can put the above equations into a matrix and attempt to generate an $\vec{x}$ such that $\|\vec{e}\|$ is as small as possible. In this case, we know that the b's are noisy, so we try to separate the b's out from the noise:

$$\vec{e} = A\vec{x} - \vec{b}. \tag{4}$$
$$\tag{5}$$

Hence, we would want to choose an $\vec{x}$ that minimizes $||\vec{e}||^2$, which is equivalent to solving the following optimization problem:

$$\min_{\vec{x}} \|\vec{e}\|^2 = \min_{\vec{x}} \left(\left\| A\vec{x} - \vec{b} \right\|^2 \right). \tag{6}$$

Let's use a simple example where $\vec{x}$ is one dimensional and $n = 2$ to illustrate this. In this case, let's use a slight abuse of notation and treat x as a scalar.

$$a_{11}x = b_1 + e_1 \tag{7}$$
$$a_{21}x = b_2 + e_2 \tag{8}$$

Hence, we have the following optimization problem:

$$\min_x [(a_{11}x - b_1)^2 + (a_{21}x - b_2)^2]. \tag{9}$$

To calculate the minimum of a function we use the idea of finding critical points, i.e. points where the slope of the function is zero. For this, we take the derivative of a function and set it equal to zero to find the critical point. Then, we can check whether the point corresponds to a maximum or minimum by using the second derivative. A positive second derivative means that it is a minimum.

If we do this, we get the following for the minimum of x in the two dimensional case:

$$x = \frac{b_1 a_{11} + b_2 a_{21}}{(a_{11})^2 + (a_{21})^2} = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|^2}$$

Note that in two dimensions, we know that the orthogonal projection is the shortest distance from a point onto a line. This principle generalizes to higher dimensions. (In 3D the shortest distance from a point to a plane is the orthogonal projection of the point onto the plane.)

Based on this strategy, we can try to generalize this to n dimensions.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{n1} & \dots & \dots & a_{nm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix} + \vec{e}$$

In order to achieve the same result as with 2 dimensions, we need $\vec{e}$ to be orthogonal to all the columns in the matrix (in order to minimize e).

$$\begin{bmatrix} a_{11} & a_{21} & \dots & a_{n1} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ \dots \\ e_n \end{bmatrix} = 0$$

This can essentially be simplified to:

$$(A^T)(\vec{e}) = 0$$

Since $\vec{e}$ is just $A\vec{x} - \vec{b}$ we have the following:

$$A^T(A\vec{x} - \vec{b}) = 0$$

$$A^T A\vec{x} - A^T\vec{b} = 0$$

$$A^T A\vec{x} = A^T\vec{b}$$

$$\vec{x} = (A^T A)^{-1} A^T \vec{b}$$

We have just derived the least squares algorithm for the general case!

Application of Least Squares

It turns out Gauss used this technique to predict where certain planets would be in their orbit. A scientist named Piazzi made 19 observations over the period of a month in regards to the orbit of Ceres (can be viewed as equations). Gauss used some of these observations. He also knew the general shape of the orbit of planets due to Kepler's laws of planetary motion. Gauss set up equations like so:

$$\alpha x^2 + \beta xy + \gamma y^2 + \delta x + \varepsilon y = \phi$$

If one divides the whole equation by ϕ, nothing significant happens so we can ignore the denominator and treat the right side of the equation as 1.

$$\alpha x^2 + \beta xy + \gamma y^2 + \delta x + \varepsilon y = 1$$

We can set up a matrix like so:

$$\begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ x_n^2 & \dots & \dots & \dots & y_n \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix}$$

Using this matrix, we can use the least squares formula to figure out the estimated value of $\begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \end{bmatrix}$ using the least square formula derived earlier:

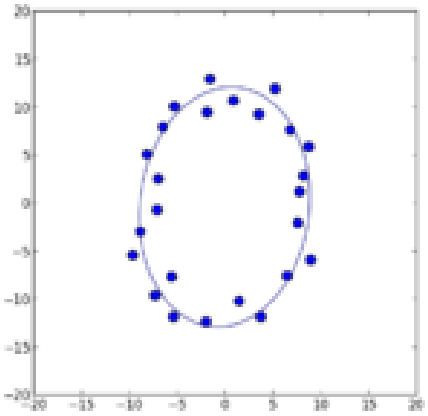
$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \end{bmatrix} = (A^T A)^{-1} A^T \begin{bmatrix} 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix} \tag{10}$$

where

$$A = \begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ x_n^2 & \dots & \dots & \dots & y_n \end{bmatrix} \tag{11}$$

.

In doing so, a possible result could be the following (the curve represents the function $\alpha x^2 + \beta xy + \gamma y^2 + \delta x + \varepsilon y = 1$):



As we can see, the least squares method is useful for fitting known equations to a curve provided that we know the general shape of the curve!