Least Squares Fitting

Least-squares fitting is common in experimental physics, engineering, and the social sciences. The typical application is where there are more constraints than variables leading to 'tall' rectangular matrices (m>n). Examples from accelerator physics include orbit control (more BPMS than correctors) and response matrix analysis (more measurements in the response matrix than variables).

The simplest linear least-squares problem can be cast in the form

$$Ax=b$$

where we look to minimize the error between the two column vectors Ax and b. The matrix A is called the *design matrix*. It is based on a linear model for the system. Column vector x contains variables in the model and column vector b contains the results of experimental measurement. In most cases, when m>n (more rows than columns) Ax does not exactly equal b, ie, b does not lie in the column space of A. The system of equations is inconsistent. The job of least-squares is to find an 'average' solution vector $\bar{x}$ that solves the system with minimum error. This section outlines the mathematics and geometrical interpretation behind linear least squares. After investigating projection of vectors into lower-dimensional subspaces, least-squares is applied to orbit correction in accelerators.

Vector Projection

We introduce least squares by way projecting a vector onto a line. From vector calculus we know the inner or 'dot' product of two vectors a and b is

$$a \cdot b = a^T b = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n = |a||b|\cos\theta$$

where θ is the angle at the vertex the two vectors. If the vertex angle is 90 degrees, the vectors are orthogonal and the inner product is zero.

Figure 1 – projection of b onto a

Referring to figure 1, the *projection* or perpendicular line from vector *b* onto the line *a* lies at point *p*. Geometrically, the point *p* is the closest point on line *a* to vector *b*. Point *p* represents the 'least-squares solution' for the 1-dimemsional projection of vector *b* into line *a*. The length of vector *b* – *p* is the error.

Defining $\bar{x}$ as the scalar coefficient that tells us how far to move along *a*, we have

$$p = \bar{x}\; a$$

Since the line between *b* and *a* is perpendicular to *a*,

$$(b - \bar{x}a) \perp a$$

so

$$a \cdot (b - \bar{x}a) = a^T (b - \bar{x}a) = 0$$

or

$$\bar{x} = \frac{a^T b}{a^T a}$$

In words, the formula reads

'take the inner product of *a* with *b* and normalize to a^2'.

The projection point *p* lies along *a* at location

$$p = \bar{x}a = \left(\frac{a^T b}{a^T a}\right)a$$

Re-writing this expression as

$$p = \left(\frac{aa^T}{a^T a}\right)b$$

isolates the *projection matrix*, $P = aa^T/a^T a$. In other words, to project vector *b* onto the line *a*, multiply '*b*' by the projection matrix to find point p=Pb. Projection matrices have important symmetry properties and satisfy $P^n=P$ – the projection of a projection remains constant.

Note that numerator of the projection operator contains the outer product of the vector '*a*' with itself. The outer product plays a role in determining how closely correlated the components of one vector are with another.

The denominator contains the inner product of a with itself. The inner provides a means to measure how parallel two vectors are ($work = force \cdot displacement$).

MATLAB Example – Projection of a vector onto a line
>>edit lsq_1

MULTI-VARIABLE LEAST SQUARES

We now turn to the multi-variable case. The projection operator looks the same but in the formulas the column vector 'a' is replaced with a matrix 'A' with multiple columns. In this case, we project *b* into the column space of A rather than onto a simple line. The goal is again to find $\bar{x}$ so as to minimize the geometric error E = $|A\bar{x} - b|^2$ where now $\bar{x}$ is a column vector instead of a single number. The quantity $A\bar{x}$ is a linear combination of the column vectors of A with coefficients $\bar{x}_1$, $\bar{x}_2$, … $\bar{x}_n$. Analogous to the single-parameter case, the least-squares solution is the point p=$A\bar{x}$ closest to point *b* in the column space of A. The error vector *b*-$A\bar{x}$ is perpendicular to that space (left null space).

The over-constrained case contains redundant information. If the measurements are not consistent or contain errors, least-squares performs an averaging process that minimizes the mean-square error in the estimate of x. If b is a vector of consistent, error-free measurements, the least-squares solution provides the exact value of x. In the less common under-constrained case, multiple solutions are possible but a solution can be constructed that minimizes the quadradic norm of x using the *pseudoinverse.*

There are several ways to look at the multi-variable least-squares problem. In each case a square coefficient matrix A^TA must be constructed to generate a set of *normal equations* prior to inversion. If the columns of A are linearly independent then A^TA is invertible and a unique solution exists for $\bar{x}$.

Figure 2 –multivariable projection

1) *Algebraic solution* – produce a square matrix and invert

$$A\bar{x} = b$$

$$A^TA\bar{x} = A^Tb \qquad \text{(normal equations for system Ax=b)}$$

$$\bar{x} = (A^TA)^{-1}A^Tb$$

The matrices A^TA and $(A^TA)^{-1}$ have far-reaching implications in linear algebra.

2) *Calculus solution* – find the minimum error

$$E^2 = |A\bar{x} - b|^2$$

$$dE^2/x = 2A^TAx - 2A^Tb = 0$$

$$A^TAx = A^Tb$$

$$\bar{x} = (A^TA)^{-1}A^Tb$$

3) *Perpendicularity*- Error vector must be perpendicular to every column vector in A

$$a_1^T(b - A\bar{x}) = 0$$

$$\ldots$$

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

or

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

or

$$A^TA\bar{x} = A^Tb$$

$$\bar{x} = (A^TA)^{-1}A^Tb$$

4) *Vector subspaces* – Vectors perpendicular to column space lie in left null space
i.e., the error vector $b - A\bar{x}$ must be in the null space of A^T

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

$$A^TA\bar{x} = A^Tb$$

$$\bar{x} = (A^TA)^{-1}A^Tb$$

MULTI-VARIABLE PROJECTION MATRICES

In the language of linear algebra, if b is not in the column space of A then Ax=b cannot be solved exactly since Ax can never leave the column space. The solution is to make the error vector Ax-b small, i.e., choose the closest point to b in the column space. This point is the *projection* of b into the column space of A.

When m > n the least-squares solution for column vector x in Ax = b is given by

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

Transforming $\bar{x}$ by matrix A yields

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

which in matrix terms expresses the construction of a perpendicular line from vector b into the column space of A. The *projection operator* P is given by

$$P = A(A^T A)^{-1} A^T \quad \sim \frac{AA^T}{A^T A}$$

Note the analogy with the single-variable case with projection operator $\frac{aa^T}{a^T a}$. In both cases, p = Pb is the component of b projected into the column space of A.

E = b – Pb is the orthogonal error vector.

> Aside: If you want to stretch your imagination, recall the SVD factorization yields V, the eigenvectors of $A^T A$, which are the axes of the error ellipsoid. The singular values are the lengths of the corresponding axes.

In orbit control, the projection operator takes orbits into orbits.

$$\bar{x} = R\theta = R(R^T R)^{-1} R^T x$$

$(R^T R)^{-1} R^T$ is a column vector of correctors, θ.

MATLAB Example – Projection of a vector into a subspace (least-squares)

```
>>edit lsq_2
```

UNDER-CONSTRAINED PROBLEMS (RIGHT PSEUDOINVERSE)

Noting that $(AA^T)(A^T A)^{-1}$=I we can write Ax=b in the form

$$Ax = (AA^T)(A^TA)^{-1}b$$

or

$$x = (A^T)(A^TA)^{-1}b = A^+b$$

where A^+b is the *right pseudoinverse* of matrix A.

MATLAB Example – Underconstrained least-squares (pseudoinverse)
>>edit lsq_3

WEIGHTED LEAST SQUARES

When individual measurements carry more or less weight, the individual rows of Ax=b can be multiplied by weighting factors.

In matrix form, weighted-least-squares looks like

$$W(Ax) = W(b)$$

where W is a diagonal matrix with the weighting factors on the diagonal. Proceeding as before,

$$(WA)^T(WA)x = (WA)^TWb$$
$$x = ((WA)^T(WA))^{-1} (WA)^TWb$$

When the weighting matrix W is the identity matrix, the equation collapses to the original solution $x = (A^TA)^{-1}A^Tb$.

In orbit correction problems, row weighting can be used to emphasize or de-emphasize specific BPMS. Column weighting can be used to emphasize or de-emphasize specific corrector magnets. In response matrix analysis the individual BPM readings have different noise factors (weights).

ORBIT CORRECTION USING LEAST-SQUARES

Consider the case of orbit correction using more BPMS than corrector magnets.

$$\mathbf{x} = \mathbf{R}\theta \qquad \text{or} \qquad \begin{bmatrix} \\ \boldsymbol{x} \\ \\ \end{bmatrix} = \begin{bmatrix} & & \\ & \boldsymbol{R} & \\ & & \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\theta} \end{bmatrix}$$

x = orbit (BPM)/constraint column vector (mm)
θ = corrector/variable column vector (ampere or mrad)
R = response matrix (mm/amp or mm/mrad)

In this case, there are more variables than constraints (the response matrix R has m>n). Using a graphical representation to demonstrate matrix dimensionality, the steps required to find a least squares solution are

$$\begin{bmatrix} \quad \boldsymbol{R}^T \quad \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x} \end{bmatrix} = \begin{bmatrix} \quad \boldsymbol{R}^T \quad \end{bmatrix} \cdot \begin{bmatrix} \quad \boldsymbol{R} \quad \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\theta} \end{bmatrix}$$

$$\begin{bmatrix} \quad \boldsymbol{R}^T \quad \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^T \boldsymbol{R} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\theta} \end{bmatrix}$$ (normal equations)

$$\begin{bmatrix} (R^T R)^{-1} \end{bmatrix} \cdot \begin{bmatrix} \quad R^T \quad \end{bmatrix} \cdot \begin{bmatrix} x \end{bmatrix} = \cdot \begin{bmatrix} \theta \end{bmatrix}$$

or

$$\boldsymbol{\theta = (R^TR)^{-1}R^Tx}$$

The projection operator predicts the orbit from corrector set θ:

$$\bar{x} = \mathbf{R(R^TR)^{-1}R^Tx}$$

and the orbit error is

$$\mathbf{e= x} - \bar{x} = \mathbf{(I - R(R^TR)^{-1}R^T)x}$$

Note that in order to correct the orbit, we reverse the sign of θ before applying the solution to the accelerator. You will not be the first or last person to get the sign wrong.

Feynman’s rule: ‘If the sign is wrong, change it’.

MATLAB Example – Least-squares orbit correction
>>edit lsq_4

RESPONSE MATRIX ANALYSIS EXAMPLE

Response matrix analysis linearizes an otherwise non-linear problem and iterates to find the solution. The linearization process amounts to a Taylor series expansion to first order. For a total of *l* quadrupole strength errors the response matrix expansion is

$$R = R_0 + \frac{\partial R_0}{\partial k_1}\Delta k_1 + \frac{\partial R_0}{\partial k_2}\Delta k_2 + ... + \frac{\partial R_0}{\partial k_l}\Delta k_l$$

$$R^{11} - R_o^{11} = \frac{\partial R_o^{11}}{\partial k_1}\Delta k_1 + .. + \frac{\partial R_o^{11}}{\partial k_l}\Delta k_l$$

where the measured response matrix R has dimensions *m x n* and all of $\{R_0, dR_o/dk_j\}$ are calculated numerically. To set up the Ax=b problem, the elements of the coefficient matrix A contain numerical derivatives dR^{ij}/dk_l. The constraint vector b has length m *times* n and contains terms from $R-R_0$. The variable vector x has length *l* and contains the Taylor expansion terms $\Delta k_1, \ldots \Delta k_l$. The matrix mechanics looks like

$$\begin{bmatrix} R^{11} - R^{11}{}_0 \\ \ldots \\ R^{1n} - R^{1n}{}_0 \\ --- \\ R^{21} - R^{21}{}_0 \\ \ldots \\ R^{2n} - R^{2n}{}_0 \\ --- \\ R^{m1} - R^{m1}{}_0 \\ \ldots \\ R^{mn} - R^{mn}{}_0 \end{bmatrix} = \left| \begin{array}{ccc} \frac{\partial R^{11}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{11}}{\partial k_l} \\ \ldots & \ldots\ \ldots & \ldots \\ \frac{\partial R^{1n}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{1n}}{\partial k_l} \\ ---- & ---- & ---- \\ \frac{\partial R^{21}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{21}}{\partial k_l} \\ \ldots & \ldots\ \ldots & \ldots \\ \frac{\partial R^{2n}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{2n}}{\partial k_l} \\ ---- & ---- & ---- \\ \frac{\partial R^{m1}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{m1}}{\partial k_l} \\ \ldots & \ldots\ \ldots & \ldots \\ \frac{\partial R^{mn}}{\partial k_1} & \ldots\ \ldots & \frac{\partial R^{mn}}{\partial k_l} \end{array} \right| \left| \begin{array}{c} \Delta k_1 \\ \ldots \\ \Delta k_l \end{array} \right|$$

The 'chi-square' fit quality factor is

$$\chi^2 = \Sigma \left(\frac{R_{ij}^{\ measure} - R_{ij}^{\ \mathrm{mod}el}}{\sigma_i} \right)^2$$

where σ_i is the rms measurement error associated with the i^{th} BPM.

SVD AND LEAST-SQUARES

The least-squares solution to $Ax=b$ where $m>n$ is given by

$$x_{lsq}= (A^TA)^{-1}A^Tb$$

Singular value decomposition of A yields

$$A = UWV^T.$$

Using the pseudoinverse,

$$A^+=VW^{-1}U^T$$

leads to

$$x_{svd} = A^+b = VW^{-1}U^T*b$$

Does $x_{lsq}= x_{svd}$ for over-constrained problems $m > n$?

Exercise: analytically substitute the singular value decomposition expressions for A and A^T to show

$$(A^TA)^{-1}A = VW^{-1}U^T.$$

Hence, SVD recovers the least-squares solution for an over-constrained system of equations.