Lecture 5: Linear Least Square Problem

NLA Chapter 11

Linear Least Square (LLS) Problem

Say you want to solve a linear system:

$$Ax = b$$

but you've got more equations than unknowns… $(m > n)$.

What do you do? A reasonable choice is to do LLS instead:

$$\min_x \ \|b - Ax\|_2$$

So instead of finding the exact solution to $Ax = b$ (which does not exist), we try to find the a best solution we can get; best means the smallest residual $b - Ax$.

(Why 2-norm?)

Pros: simple computation, easy statistical/geometric interpretation, long history…

Cons: sensitive to outliers

LLS in the language of linear algebra

$Ax = b$ with $A \in \mathbb{R}^{m \times n}, m > n$ in general does not have solution.

What this means is that in general, b is not in the range(A), for this overdetermined system.

LLS problem seeks to minimize the residual

$$r = b - Ax \in \mathbb{R}^m$$

If we measure the size of residual in 2-norm (corresponds to Euclidean distance), then we are seeking a vector $x \in \mathbb{R}^n$ such that Ax is the closest point in range(A) to b.

Example: Polynomial Data-Fitting

Polynomial interpolation: square system; linear solve;

Polynomial data-fitting (low degree): rectangular system: LLS.

Polynomial Interpolation

We are given m distinct points $x_1, \dots, x_m \in \mathbb{R}$ and data $y_1, \dots, y_m \in \mathbb{R}$ at these points. Then there exists a unique polynomial interpolant to these data in these points, that is, a polynomial of degree at most $m - 1$

$$p(x) = c_0 + c_1 x + \cdots + c_{m-1} x^{m-1},$$

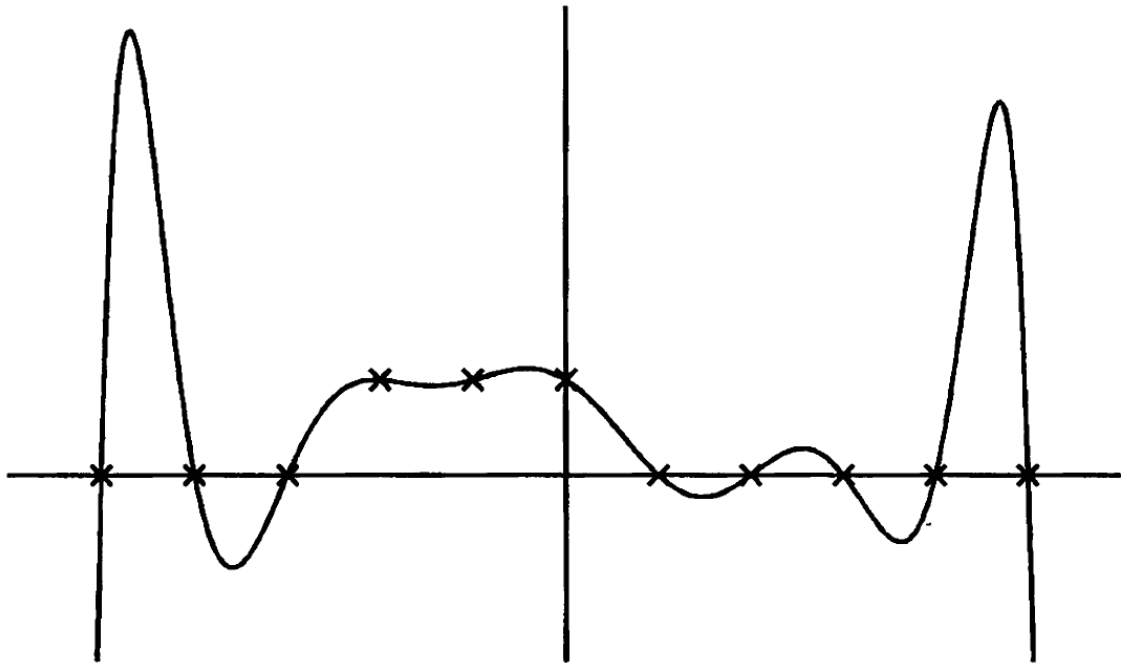
with the property that $p(x_i) = y_i, i = 1, \dots, m$. The relationship of the data $\{x_i\}, \{y_i\}$ to the coefficients $\{c_i\}$ can be expressed by the square Vandermonde system:

$$\begin{bmatrix} 1 & x_1 & x_1^2 & & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{m-1} \\ 1 & x_3 & x_3^2 & & x_3^{m-1} \\ & \vdots & & & \vdots \\ 1 & x_m & x_m^2 & \cdots & x_m^{m-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{m-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}.$$

The square Vandermonde system can be solved as long as the $\{x_i\}$ are distinct.

In the figure we have 11 discrete data points (supposedly coming from square wave) and its degree 10 polynomial interpolant.

Is this a reasonable reflection of the data?



The phenomenon that the polynomial interpolant is often not a good data fit is quite typical. (The polynomial interpolant oscillates too much towards the extremal data points)

And to make it even worse, the more data points you use, the less fit the interpolant is (more extreme oscillation!)

Even when the fit happens to be good, the process of the interpolation may be ill-conditioned, (sensitive to the perturbations in the data). Choosing nonuniformly spaced $\{x_i\}$ may help (e.g. Chebyshev points) but is not always possible in application.

So, is there a better way?

Polynomial Least Square Fitting

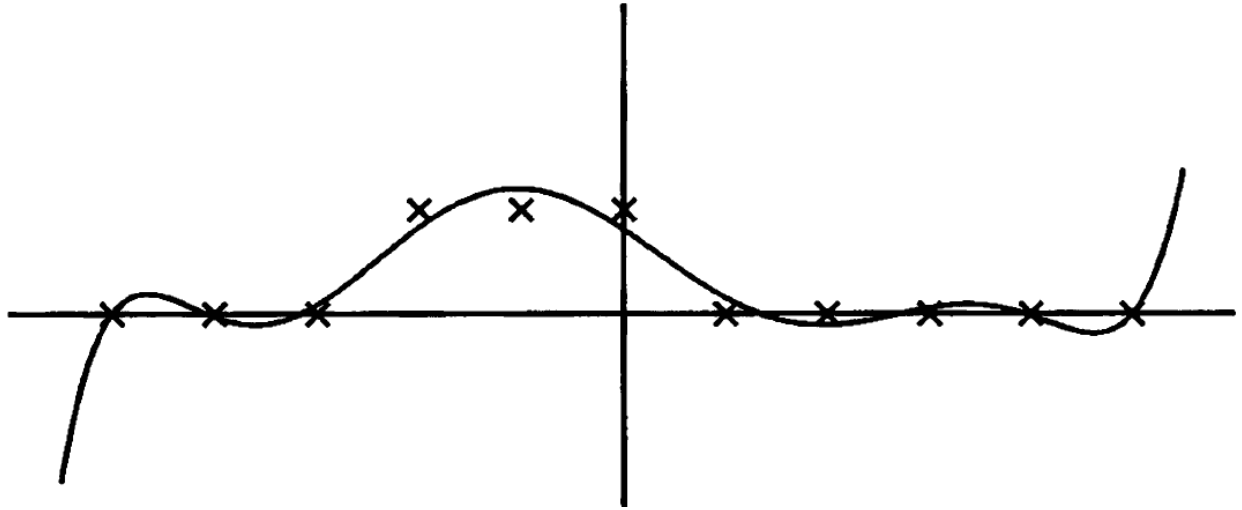
We can do better by reducing the degree of the polynomial to $n - 1$ $(n < m)$. Such a polynomial is a least square fit to the data if it minimizes the sum of squares of deviations from the data, i.e.:

$$\sum_{i=1}^{m} (p(x_i) - y_i)^2$$

The sum of squares is equal to the 2-norm of residuals $\|r\|_2^2$, for the **rectangular** Vandermonde system:

$$\begin{bmatrix} 1 & x_1 & & x_1^{n-1} \\ 1 & x_2 & \cdots & x_2^{n-1} \\ 1 & x_3 & & x_3^{n-1} \\ & \vdots & & \vdots \\ 1 & x_m & \cdots & x_m^{n-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} \approx \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}.$$

If we fit the same 11 data points with a degree 7 polynomial:



The polynomial no longer interpolates all the points, but the fit seems more reasonable. Plus, it's less sensitive to the perturbations in the data.

How to compute this polynomial data-fit?

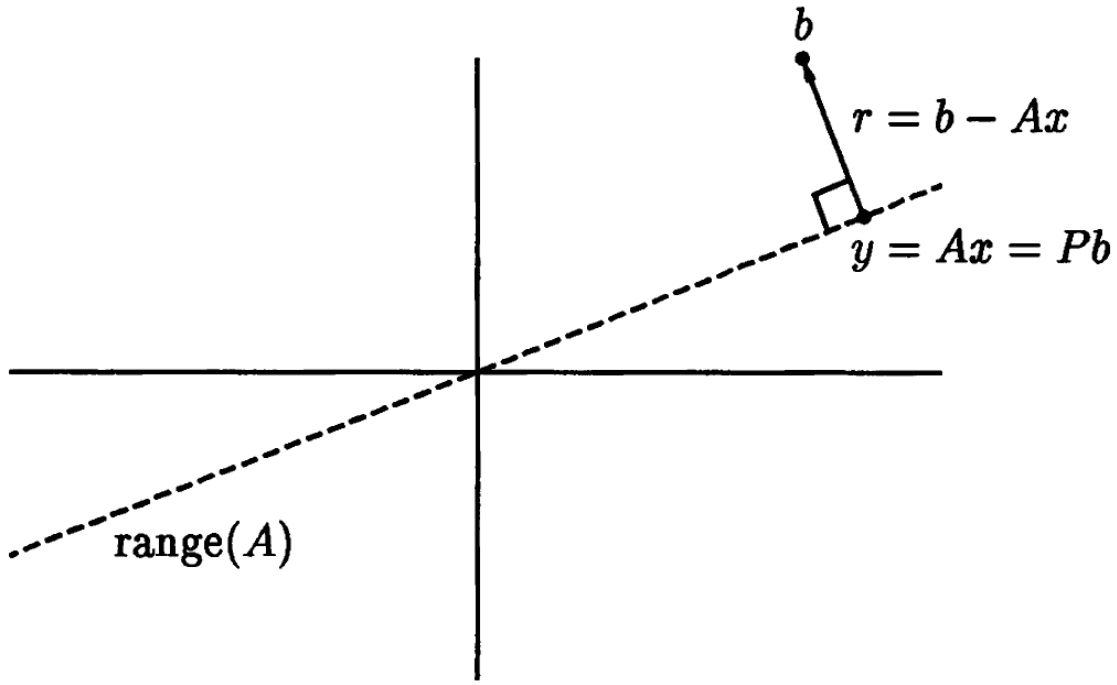
Orthogonal Projection

How to solve least square problem?

Geometrically, the closest point Ax to b happens iff $Ax = Pb$ where P is some **orthogonal** projector onto range(A).

I.e., the residual is orthogonal to range(A): $r \perp \text{range}(A)$

I.e. algebraically, $A^T r = 0$.



Normal Equations

From last slide we have optimality condition:

$$A^T r = 0$$

or,

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

This square (!) system is called Normal Equations. It uniques determines x assuming full rank of A.

Another equivalent way of saying is:

$$Pb = Ax$$

where P is **the** orthogonal projector onto range(A).

Pseudo Inverse

From the normal equations, the solution to LLS problem is given by

$$x = \underbrace{(A^T A)^{-1} \; A^T}_{\text{psuedo inverse of } A} \quad b$$

The Pseudo Inverse of A is defined as

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

The pseudo inverse maps a vector $b \in \mathbb{R}^m$ to vector $x \in \mathbb{R}^n$. I.e., the shape of A^+ is $n \times m$ (like the transpose of A), which has more columns than rows.

Okay, to summarize, solution of LLS problem is given by:

$$x = A^+ b$$

If you want y then solve $y = Pb$, where P is ortho projector onto range(A).

Pseudo Inverse

In terms of SVD, the pseudo inverse is like this:

$$A = U\Sigma V^T$$
$$A^+ = V\Sigma^{-1}U^T$$

(please verify this)

If $m \geq n$, then $A^+A = I$, so A^+ is the left inverse of A.

(What about AA^+? Hint: does it look like a projector)

No matter the shape of A, we have the following (check this!):

$$AA^+A = A$$
$$A^+AA^+ = A^+$$

Three Algorithms to Solve LLS

1. Normal Equations: $A^T Ax = A^T b$
2. QR factorization: $A = \hat{Q}\hat{R}, P = \hat{Q}\hat{Q}^T$
3. SVD: $A = \hat{U}\Sigma V^T, P = \hat{U}\hat{U}^T$

Let's have a little competition of these three players, shall we…

The criteria:

- Stability (sensitivity to perturbations)
- Cost (number of FLOPs)

Normal Equations

The classical way (or the obvious one?) to solve LLS problem is through normal equations.

If A has full rank, $A^T A$ is non-singular square matrix, and additionally it's positive definite. The standard method of solving symmetric positive definite (SPD) system is by Cholesky factorization: $A^T A = R^T R$.

Algorithm 11.1. Least Squares via Normal Equations

1. Form the matrix A^*A and the vector A^*b. mn^2
2. Compute the Cholesky factorization $A^*A = R^*R$. $\frac{1}{3}n^3$
3. Solve the lower-triangular system $R^*w = A^*b$ for w. $O(n^2)$
4. Solve the upper-triangular system $Rx = w$ for x. $O(n^2)$

Total FLOPs:

$\sim mn^2 + \frac{1}{3}n^3$

QR Factorization

The modern classical way to solve LLS problem is through QR factorization: $A = \hat{Q}\hat{R}$. The ortho projector onto range(A) is therefore $P = \hat{Q}\hat{Q}^T$. So we have:

$$y = Pb = \hat{Q}\hat{Q}^T b$$

Taken into account $y = Ax = \hat{Q}\hat{R}x$, we have the equation:

$$\hat{Q}\hat{Q}^T b = \hat{Q}\hat{R}x$$

Multiply both sides by $\hat{Q}^T$, we get (convince your self of this!)

$$\hat{Q}^T b = \hat{R}x$$

Algorithm 11.2. Least Squares via QR Factorization

1. Compute the reduced QR factorization $A = \hat{Q}\hat{R}$. $2mn^2 - \frac{2}{3}n^3$
2. Compute the vector $\hat{Q}^* b$. $O(n^2)$
3. Solve the upper-triangular system $\hat{R}x = \hat{Q}^* b$ for x. $O(n^2)$

Total Cost:

$\sim 2mn^2 - \frac{2}{3}n^3$

SVD

If we have SVD $A = \hat{U}\Sigma V^T$, just like in QR we have $P = \hat{U}\hat{U}^T$. From $y = Ax$ we have:

$$\hat{U}\Sigma V^T x = \hat{U}\hat{U}^T b$$

Multiply both sides by $\hat{U}^T$:

$$\Sigma V^T x = \hat{U}^T b$$

Algorithm 11.3. Least Squares via SVD

1. Compute the reduced SVD $A = \hat{U}\hat{\Sigma}V^*$.
2. Compute the vector $\hat{U}^*b$.
3. Solve the diagonal system $\hat{\Sigma}w = \hat{U}^*b$ for w.
4. Set $x = Vw$.

cost of SVD??

Typical Cost:

$2mn^2 + 11n^3$

Comparison

When the matrix A is ill-conditioned (for now, intuitively we can think of A being nearly rank-deficient), then more stable algorithm tend to work better.

We'll see what "condition" and "stability" exactly mean in the next set of lectures. You will play with it in the HW1, so see for yourself.

References

Numerical Linear Algebra, Trefethen & Bau, Chapter 11

Gallier, J., & Quaintance, J. (2017). *Fundamentals of Linear Algebra and Optimization*. https://doi.org/10.1007/978-1-4419-9887-3