CS 4220: Numerical Analysis

Introduction to least squares

David Bindel

2026-02-20

Least squares: the big idea

Least squares problems are a special sort of minimization problem. Suppose $A \in \mathbb{R}^{m \times n}$ where $m > n$. In general, we cannot solve the *overdetermined* system $Ax = b$; the best we can do is minimize the *residual* $r = b - Ax$. In the least squares problem, we minimize the two norm of the residual:

$$\text{Find } x \text{ to minimize } \|r\|_2^2 = \langle r, r \rangle.$$

This is not the only way to approximately solve the system, but it is attractive for several reasons:

1. It's mathematically attractive: the solution of the least squares problem is $x = A^\dagger b$ where $A^\dagger$ is the *Moore-Penrose pseudoinverse* of A.
2. There's a nice picture that goes with it — the least squares solution is the projection of b onto the range of A, and the residual at the least squares solution is orthogonal to the range of A.
3. It's a mathematically reasonable choice in statistical settings when the data vector b is contaminated by Gaussian noise.

Cricket chirps: an example

Did you know that you can estimate the temperature by listening to the rate of chirps? The data shown in Figure 1 represents measurements of the number of chirps (over 15 seconds) of a striped ground cricket at different temperatures measured in degrees Farenheit[1]. The plot

[1] Data set at https://www.mathbits.com/MathBits/TISection/Statistics2/linearREAL.html

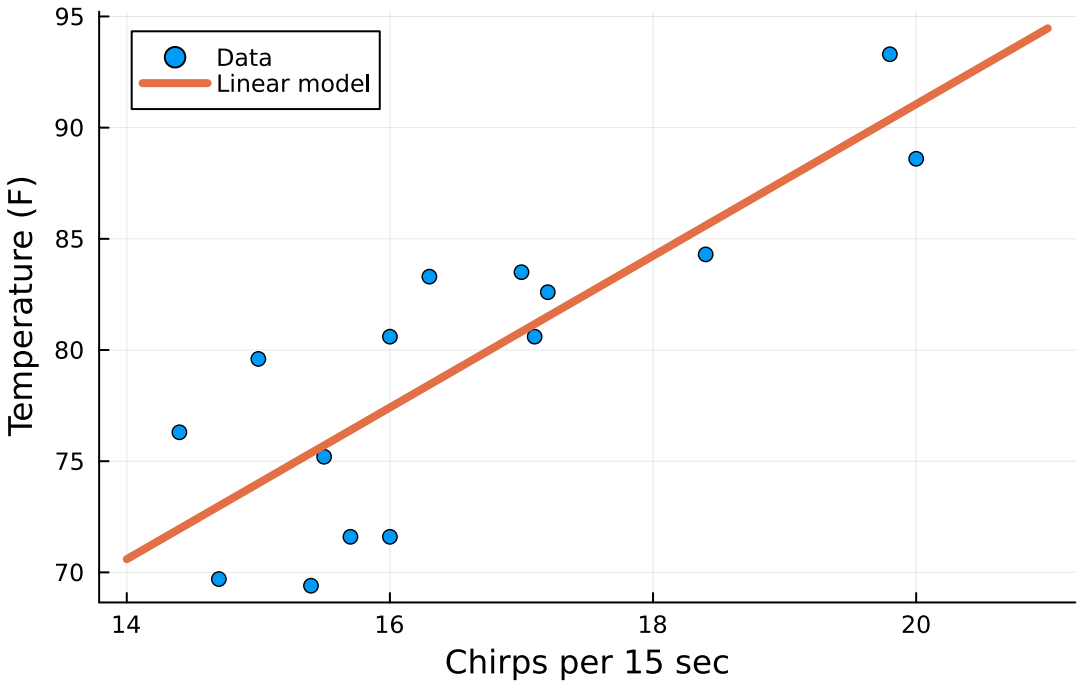

Figure 1: Cricket measurements and least-squares fit

shows that the two are roughly correlated: the higher the temperature, the faster the crickets chirp. We can quantify this by attempting to fit a linear model

$$\text{temperature} = \alpha \cdot \text{chirps} + \beta + \epsilon$$

where ϵ is an error term. To solve this problem by linear regression, we minimize the residual

$$r = b - Ax$$

where

$$\begin{aligned} b_i &= \text{temperature in experiment } i \\ A_{i1} &= \text{chirps in experiment } i \\ A_{i2} &= 1 \\ x &= \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \end{aligned}$$

Julia is capable of solving least squares problems using the backslash operator; that is, if `chirps` and `temp` are column vectors in Julia, we can solve this regression problem as

```
A = [chirps ones(ndata)]
x = A\temp
```

The algorithms underlying that backslash operation will make up most of the next lecture.

In more complex examples, we want to fit a model involving more than two variables. This still leads to a linear least squares problem, but one in which A may have more than one or two columns. As we will see later in the semester, we also use linear least squares problems as a building block for more complex fitting procedures, including fitting nonlinear models and models with more complicated objective functions.

Normal equations

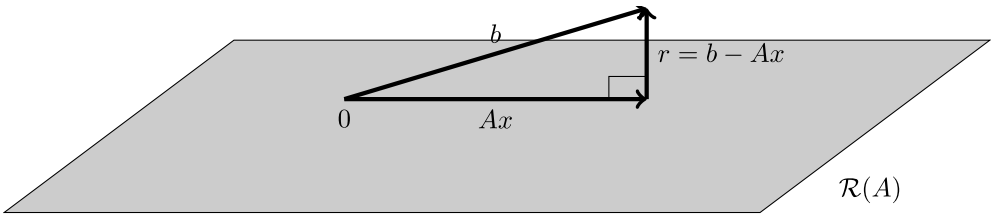

Figure 2: Picture of a linear least squares problem. The vector Ax is the closest vector in $\mathcal{R}(A)$ to a target vector b in the Euclidean norm. Consequently, the residual $r = b - Ax$ is normal (orthogonal) to $\mathcal{R}(A)$.

When we minimize the Euclidean norm of $r = b - Ax$, we find that r is *normal* to everything in the range space of A (Figure 2):

$$b - Ax \perp \mathcal{R}(A),$$

or, equivalently, for all $z \in \mathbb{R}^n$ we have

$$0 = (Az)^T(b - Ax) = z^T(A^Tb - A^TAx).$$

The statement that the residual is orthogonal to everything in $\mathcal{R}(A)$ thus leads to the *normal equations*

$$A^TAx = A^Tb.$$

To see why this is the right system, suppose x satisfies the normal equations and let $y \in \mathbb{R}^n$ be arbitrary. Using the fact that $r \perp Ay$ and the Pythagorean theorem, we have

$$\|b - A(x + y)\|^2 = \|r - Ay\|^2 = \|r\|^2 + \|Ay\|^2 > 0.$$

The inequality is strict if $Ay \neq 0$; and if the columns of A are linearly independent, $Ay = 0$ is equivalent to $y = 0$.

We can also reach the normal equations by calculus. Define the least squares objective function:

$$F(x) = \|Ax - b\|^2 = (Ax - b)^T(Ax - b) = x^TA^TAx - 2x^TA^Tb + b^Tb.$$

The minimum occurs at a *stationary point*; that is, for any perturbation δx to x we have

$$\delta F = 2\delta x^T (A^T A x - A^T b) = 0;$$

equivalently, $\nabla F(x) = 2(A^T A x - A^T b) = 0$ — the normal equations again!

A family of factorizations

Cholesky

If A is full rank, then $A^T A$ is symmetric and positive definite matrix, and we can compute a Cholesky factorization of $A^T A$:

$$A^T A = R^T R.$$

The solution to the least squares problem is then

$$x = (A^T A)^{-1} A^T b = R^{-1} R^{-T} A^T b,$$

or, in Julia world

```
AC = cholesky(A'*A)
x = AC\(A'*b)  # Using the factorization object, OR
x = AC.U\(AC.U'\(A'*b))
```

Economy QR

The Cholesky factor R appears in a different setting as well. Let us write $A = QR$ where $Q = AR^{-1}$; then

$$Q^T Q = R^{-T} A^T A R^{-1} = R^{-T} R^T R R^{-1} = I.$$

That is, Q is a matrix with orthonormal columns. This "economy QR factorization" can be computed in several different ways, including one that you have seen before in a different guise (the Gram-Schmidt process). Julia provides a numerically stable method to compute the QR factorization via

```
AC = qr(A)
```

and we can use the QR factorization directly to solve the least squares problem without forming $A^T A$ by

```
AC = qr(A,0)
x = AC\b   # Using the factorization object, OR
x = AC.R\((AC.Q'*b)[1:m])
```

Full QR

There is an alternate "full" QR decomposition where we write

$$A = QR, \text{ where } Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \in \mathbb{R}^{m \times m}, R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{m \times n}.$$

To see how this connects to the least squares problem, recall that the Euclidean norm is invariant under orthogonal transformations, so

$$\|r\|^2 = \|Q^T r\|^2 = \left\| \begin{bmatrix} Q_1^T b \\ Q_2^T b \end{bmatrix} - \begin{bmatrix} R_1 \\ 0 \end{bmatrix} x \right\|^2 = \|Q_1^T b - R_1 x\|^2 + \|Q_2^T b\|^2.$$

We can set $\|Q_1^T v - R_1 x\|^2$ to zero by setting $x = R_1^{-1} Q_1^T b$; the result is $\|r\|^2 = \|Q_2^T b\|^2$.

The actual thing computed by Julia is a sort of hybrid of the full and economy decompositions. The data structure representing Q (in compressed form) can reconstruct the full orthogonal matrix; but the R factor is stored as in the economy form.

SVD

The full QR decomposition is useful because orthogonal transformations do not change lengths. Hence, the QR factorization lets us change to a coordinate system where the problem is simple without changing the problem in any fundamental way. The same is true of the SVD, which we write as

$$\begin{aligned} A &= \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^T && \text{Full SVD} \\ &= U_1 \Sigma V^T && \text{Economy SVD.} \end{aligned}$$

As with the QR factorization, we can apply an orthogonal transformation involving the factor U that makes the least squares residual norm simple:

$$\|U^T r\|^2 = \left\| \begin{bmatrix} U_1^T b \\ U_2^T b \end{bmatrix} - \begin{bmatrix} \Sigma V^T \\ 0 \end{bmatrix} x \right\| = \|U_1^T b - \Sigma V^T x\|^2 + \|U_2^T b\|^2,$$

and we can minimize by setting $x = V \Sigma^{-1} U_1^T b$.

The Moore-Penrose pseudoinverse

If A is full rank, then A^TA is symmetric and positive definite matrix, and the normal equations have a unique solution

$$x = A^\dagger b \text{ where } A^\dagger = (A^TA)^{-1}A^T.$$

The matrix $A^\dagger \in \mathbb{R}^{n\times m}$ is the *Moore-Penrose pseudoinverse*. We can also write $A^\dagger$ via the economy QR and SVD factorizations as

$$A^\dagger = R^{-1}Q_1^T,$$
$$A^\dagger = V\Sigma^{-1}U_1^T.$$

If $m = n$, the pseudoinverse and the inverse are the same. For $m > n$, the Moore-Penrose pseudoinverse has the property that

$$A^\dagger A = I;$$

and

$$\Pi = AA^\dagger = Q_1Q_1^T = U_1U_1^T$$

is the *orthogonal projector* that maps each vector to the closest vector (in the Euclidean norm) in the range space of A.

The good, the bad, and the ugly

At a high level, there are two pieces to solving a least squares problem:

1. Project b onto the span of A.
2. Solve a linear system so that Ax equals the projected b.

Consequently, there are two ways we can get into trouble in solving least squares problems: either b may be nearly orthogonal to the span of A, or the linear system might be ill conditioned.

Let's first consider the issue of b nearly orthogonal to the range of A first. Suppose we have the trivial problem

$$A = \begin{bmatrix}1\\0\end{bmatrix}, \quad b = \begin{bmatrix}\epsilon\\1\end{bmatrix}.$$

The solution to this problem is $x = \epsilon$; but the solution for

$$A = \begin{bmatrix}1\\0\end{bmatrix}, \quad \hat{b} = \begin{bmatrix}-\epsilon\\1\end{bmatrix}$$

is $\hat{x} = -\epsilon$. Note that $\|\hat{b} - b\|/\|b\| \approx 2\epsilon$ is small, but $|\hat{x} - x|/|x| = 2$ is huge. That is because the projection of b onto the span of A (i.e. the first component of b) is much smaller than b itself;

so an error in b that is small relative to the overall size may not be small relative to the size of the projection onto the columns of A.

Of course, the case when b is nearly orthogonal to A often corresponds to a rather silly regressions, like trying to fit a straight line to data distributed uniformly around a circle, or trying to find a meaningful signal when the signal to noise ratio is tiny. This is something to be aware of and to watch out for, but it isn't exactly subtle: if $\|r\|/\|b\|$ is near one, we have a numerical problem, but we also probably don't have a very good model. A more subtle problem occurs when some columns of A are nearly linearly dependent (i.e. A is ill-conditioned).

The *condition number of A for least squares* is

$$\kappa(A) = \|A\|\|A^\dagger\| = \sigma_1/\sigma_n.$$

If $\kappa(A)$ is large, that means:

1. Small relative changes to A can cause large changes to the span of A (i.e. there are some vectors in the span of $\hat{A}$ that form a large angle with all the vectors in the span of A).
2. The linear system to find x in terms of the projection onto A will be ill-conditioned.

If θ is the angle between b and the range of A, then the sensitivity to perturbations in b is

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\kappa(A)}{\cos(\theta)} \|\delta b\| \|b\|$$

while the sensitivity to perturbations in A is

$$\frac{\|\delta x\|}{\|x\|} \leq \left(\kappa(A)^2 \tan(\theta) + \kappa(A)\right) \frac{\|\delta A\|}{\|A\|}$$

Even if the residual is moderate, the sensitivity of the least squares problem to perturbations in A (either due to roundoff or due to measurement error) can quickly be dominated by $\kappa(A)^2 \tan(\theta)$ if $\kappa(A)$ is at all large.

In regression problems, the columns of A correspond to explanatory factors. For example, we might try to use height, weight, and age to explain the probability of some disease. In this setting, ill-conditioning happens when the explanatory factors are correlated — for example, perhaps weight might be well predicted by height and age in our sample population. This happens reasonably often. When there is a lot of correlation, we have an *ill-posed* problem; we will talk about this case in a couple lectures.