CME 104: Pseudoinverse

Enze Chen

Date: April 12, 2018

1 Motivation

So far we've worked with square matrices and fat matrices (more columns than rows), but there are many times where we will encounter a *skinny matrix*, or one with more rows than columns. For example, we might be trying to solve a system of equations $A\vec{x} = \vec{b}$ where we have more equations (constraints) than variables (unknowns). Such a system is called **overdetermined**, and even though this system cannot be solved exactly (A is not invertible), we can try to find an *approximate solution* that minimizes the error we incur.

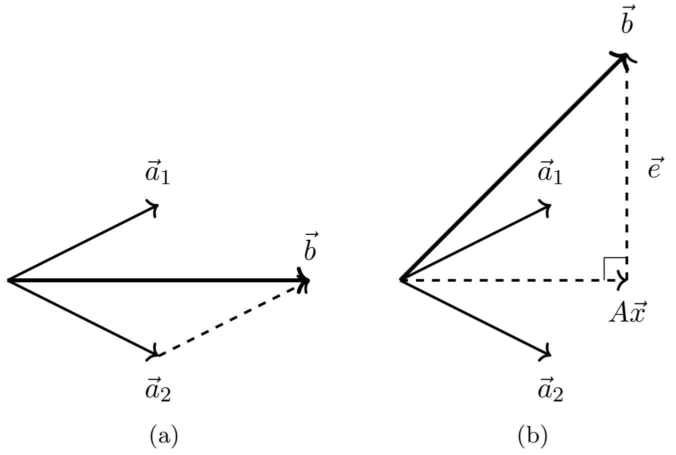

Figure 1: (a) 2D example where $\vec{b}$ can be expressed as a linear combination of $\vec{a}_1$ and $\vec{a}_2$, i.e. $\vec{b}$ is in the column space of A. (b) $\vec{b}$ is no longer in the column space of A, but we can find the vector $A\vec{x}$ such that the error $\vec{e}$ is minimized.

More generally, when we say that $A\vec{x} = \vec{b}$ has a solution, what we really mean is that the vector $\vec{b}$ lies in the column space of A. If we split the matrix A up into its columns and let each x_i be coefficients, we see that

$$A\vec{x} = \sum_{i=1}^{n} x_i \cdot \vec{a}_i = \vec{b}$$

and therefore a solution to this system exists only if $\vec{b}$ can be expressed as a linear combination of the columns of A. Figure 1(a) gives us a visualization for what's going on in the 2D case.

Of course, there are many times where $\vec{b}$ is unfortunately not in the column space of A, and we must find the best approximate solution instead of an exact solution (which doesn't exist). How do we quantify "best?" Let's look at Figure 1(b) which shows how $\vec{b}$ extends out of the plane defined by $\vec{a}_1$ and $\vec{a}_2$. What we can do now is find the vector $\vec{x}$ such that the error between $A\vec{x}$ and $\vec{b}$, represented by the vector $\vec{e}$, minimized. We call $A\vec{x}$ the *projection* of $\vec{b}$ in the column space of A, and note that $\vec{e}$ is minimized when it is orthogonal to the plane. Now if we have a method of solving for $\vec{x}$ when $\vec{e}$ is orthogonal to the column space of A, then this will be the best approximate solution that minimizes error.

2 Approach

From Figure 1(b), we can clearly see that

$$\vec{e} = \vec{b} - A\vec{x} \tag{1}$$

Recall that we claimed above that $\vec{e}$ is minimized when it is orthogonal to the column space of A, and recall that when two vectors are orthogonal, their dot product is zero. This means that the dot product between each column of A with $\vec{e}$ is equal to zero, i.e.

$$\begin{aligned} \vec{a}_1^T\vec{e} &= 0 \\ \vec{a}_2^T\vec{e} &= 0 \\ &\vdots \\ \vec{a}_n^T\vec{e} &= 0 \end{aligned}$$

If we combine all the equations above into a single matrix-vector expression, we note that writing all of the columns of A as row vectors and stacking them will result in the *transpose* of A, or A^T. Thus the above equations can be written more concisely as $A^T\vec{e} = \vec{0}$.

Now we can take equation 1 and multiply both sides on the left by A^T to obtain

$$\begin{aligned} A^T\vec{e} &= A^T(\vec{b} - A\vec{x}) \\ \vec{0} &= A^T\vec{b} - A^TA\vec{x} \\ A^TA\vec{x} &= A^T\vec{b} \\ &\boxed{\vec{x} = (A^TA)^{-1}A^T\vec{b}} \end{aligned} \tag{2}$$

This is the formula that we can use to obtain a vector $\vec{x}$ such that $A\vec{x}$ is the vector in the column space of A that is the "best" approximation to $\vec{b}$ in the sense that the error is minimized. Note that this equation appears similar to our previous expression for solving systems using the inverse, $\vec{x} = A^{-1}\vec{b}$, except instead of A^{-1} we have $(A^TA)^{-1}A^T$. This last matrix and equation 2 are so popular and widely used that we give this matrix a special name. We call $(A^TA)^{-1}A^T$ the **pseudoinverse**[1] and let the symbol $A^\dagger = (A^TA)^{-1}A^T$ (pronounced "A-dagger") represent this matrix. Note that if A was an invertible square matrix, we have

$$A^\dagger = (A^TA)^{-1}A^T = A^{-1}(A^T)^{-1}A^T = A^{-1}$$

3 Application: Least squares fitting

Let's see an application of the pseudoinverse in action, in what is perhaps the most canonical setting for the pseudoinverse: solving the least squares problem. Say we're given a set of (x_i, y_i) data points and wish to find the line of best fit that approximates the data (Figure 2). We know that the equation for the line is given by $y = ax + b$, but we also know that one set of (a, b) values cannot give us the exact solution to each pair of (x_i, y_i), so we approximate.

[1] Depending on the community and textbook, you'll also hear this called the "Moore-Penrose inverse."

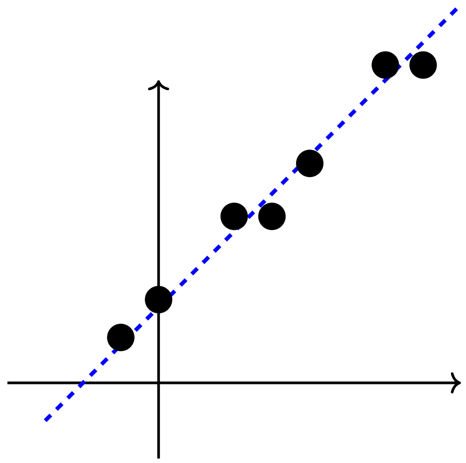

Figure 2: The least squares problem. We're given a set of data points and want to find the line of best fit to approximate the data points while minimizing the residual error (vertical distance between each point and the line).

We start by explicitly writing each equation:

$$\begin{aligned} y_1 &= ax_1 + b = \begin{bmatrix} x_1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} \\ y_2 &= ax_2 + b = \begin{bmatrix} x_2 & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} \\ &\vdots \\ y_m &= ax_m + b = \begin{bmatrix} x_m & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} \end{aligned}$$

then stack all the equations to get the matrix-vector equation:

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{\vec{y}} = \underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_m & 1 \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} a \\ b \end{bmatrix}}_{\vec{u}}$$

Now we see that we have a $X\vec{u} = \vec{y}$ equation for an overdetermined system and X is skinny. Therefore, we can find the approximate solution to this system of equations using the pseudoinverse,

$$\vec{u} = (X^T X)^{-1} X^T \vec{y}$$

Food for thought: When does it fail?

Someone in class today asked a great question about when the pseudoinverse might not even exist, and in general it is a good idea to ask this question whenever you encounter a new algorithm for solving a problem. In the same way that the inverse might not exist for certain matrices, we might not actually be able to find the inverse of A^TA. It turns out that *if* A is full column rank, i.e. $\text{rank}(A) = n$, then A^TA is *always* invertible. I'll point to Stack Exchange for the proof (it's not bad), but we can reason a bit about this and see that *in most cases*, A^TA will be invertible. In typical least-squares applications, for example, we will have far more data points (rows of the matrix) than the number of variables/dimensions (columns of the matrix). If I give you just a handful of column vectors, say 10, where each one is very long, say 10,000 elements (so your matrix is 10,000 by 10), then it's not hard to see that it is statistically very unlikely for these 10 vectors to be linearly dependent such that somehow all 10,000 index positions form the same linear combination. Therefore, there's a high probability that a skinny matrix A is full column rank and the pseudoinverse will exist for most applications.