1 Matrix Multiplication (Section 1.1)

Consider an $n \times m$ matrix, or a matrix with n rows and m columns:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}.$$

The entries of A might be real or complex numbers although, for now, we'll assume they're real. Suppose we have an m-tuple (or vector) of real numbers:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}.$$

1.1 Multiplying Matrix by Vector

Suppose A is an $n \times \boxed{m}$ matrix and $\mathbf{x}$ is a vector with m elements (i.e., a $\boxed{m} \times 1$ matrix). Let's suppose we wanted to find $A\mathbf{x}$. There are two ways we can do this.

1.1.1 Solving as a Single Component

We can solve for

$$A\mathbf{x} = \mathbf{b},$$

where

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

Here,

$$b_i = a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{im}x_m = \sum_{j=1}^{m} a_{ij}x_j.$$

(Example.) Consider

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix}.$$

We note that

$$A\mathbf{x} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

where

$$b_1 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = 1(7) + 2(8) + 3(9) = 50$$

and

$$b_2 = a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = 4(7) + 5(8) + 6(9) = 122.$$

We can write some code to perform this operation for us:

$\mathbf{b} \leftarrow \mathbf{0}$
for $i = 1, \dots, n$ **do**
  for $i = 1, \dots, m$ **do**

$b_i \leftarrow b_i + a_{ij}x_j$
end for
end for

Note that the j-loop accumulates the inner product b_i.

1.1.2 Solving as a Formula

We can also solve for $A\mathbf{x} = \mathbf{b}$ by considering the following:

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} x_1 + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{bmatrix} x_2 + \ldots + \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix} x_m.$$

This shows that $\mathbf{b}$ is a linear combination of the columns of A.

(Example.) Consider

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix}.$$

Using this approach, we now have

$$\begin{bmatrix} 1 \\ 4 \end{bmatrix} 7 + \begin{bmatrix} 2 \\ 5 \end{bmatrix} 8 + \begin{bmatrix} 3 \\ 6 \end{bmatrix} 9 = \begin{bmatrix} 7 \\ 28 \end{bmatrix} + \begin{bmatrix} 16 \\ 40 \end{bmatrix} + \begin{bmatrix} 27 \\ 54 \end{bmatrix} = \begin{bmatrix} 50 \\ 122 \end{bmatrix}.$$

Proposition. *If $\mathbf{b} = A\mathbf{x}$, then $\mathbf{b}$ is a linear combination of the columns of A. If we let A_j denote the jth column of A, we have*

$$\mathbf{b} = \sum_{j=1}^{m} A_j x_j.$$

We can also express this as pseudocode:

$\mathbf{b} \leftarrow \mathbf{0}$
for $j = 1, \ldots, m$ **do**
$\mathbf{b} \leftarrow \mathbf{b} + A_j x_j$
end for

If we use a loop to perform each vector operation, the code becomes:

$\mathbf{b} \leftarrow \mathbf{0}$
for $j = 1, \ldots, m$ **do**
for $i = 1, \ldots, n$ **do**
$b_i \leftarrow b_i + a_{ij}x_j$
end for
end for

Notice how the loops for rows and columns are interchangeable.

1.2 Flop Counts

We note that real numbers are normally stored in computers in a floating-point format. The arithmetic operations that a computer performs on these numbers are called *floating-point operations*, or *flops*. So,

$$b_i \leftarrow b_i + a_{ij}x_j$$

involves two flops: one floating-point multiply and one floating-point add. **Essentially**, we would like to count the number of operations on real numbers, or the number of addition, subtraction, multiplication, and division operations, within our program.

(Example.) Consider

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}.$$

Let's compute the inner product[a]:

$$\begin{aligned} \langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^T \cdot \mathbf{w} &= \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \\ &= v_1 \cdot w_1 + v_2 \cdot w_2 + \dots + v_n \cdot w_n \\ &= \sum_{i=1}^{n} v_i w_i. \end{aligned}$$

There are $n - 1$ addition operations and n multiplication operations, for a total of $2n - 1$ flop count.

[a] A generalization of the dot product

1.2.1 Big-O Notation

We note that $\mathbf{v}^T\mathbf{w}$, where $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$ needs $2n$ flops. This is the same thing as saying that the operation takes $\mathcal{O}(n)$ time as $n \mapsto \infty$. This means that we can disregard the implied constant, which is 2 in this case. The reason why we can do this is because, as n grows larger, the constant doesn't really matter all that much.

For instance, if n is large, then $\mathcal{O}(n)$ is faster than $\mathcal{O}(n^2)$, and $\mathcal{O}(n^2)$ is faster than $\mathcal{O}(n^3)$.