Efficiency of Elimination

Learning Goal: to obtain an operation count on elimination and compare with other methods of solving systems.

We’ve spoken many times about the efficiency of elimination. Let’s see what that means. We will actually count the number of operations we have to do in order to row reduce and solve systems of equations.

Let’s count each basic arithmetic operation as one step. So any addition, subtraction, multiplication, or division of real numbers counts one step. This isn’t strictly fair, since usually a computer will take slightly longer to multiply two numbers than to add them, but if we take the length of the longest operation as the basic unit of time we will certainly overshoot the true amount of time it takes, and coming in under budget is always good policy!

The row reduction phase

We will keep separate counts for the left-hand and right-hand sides, because we will want to compare the elimination to the forward substitution method of finding the right-hand side used for back substitution.

Consider eliminating the entry below the first pivot. We first have to find the multiplier—\(a_{21}/a_{11}\). This is one operation. Now we know that the first entry in the second row will become zero, so there is no operation needed to find this new entry. But every other entry in the second row requires two operations. We must multiply the entry in the first row times the multiplier, then subtract it from the entry in the second row. Since there are \(n-1\) other entries in the row, we get a total of \(2(n-1)\) operation to perform this elimination step. In addition, there are two operations needed to adjust the second entry of the right-hand side (another multiply-subtract combo).

This must be repeated on each of the third, fourth, …, \(n\)th rows. Thus we have \(2n-1\) operations per row, times \(n-1\) rows, giving \(2n^2 - 3n + 1\) operations. There are \(2n-2\) operations needed on the right-hand side. Now the first column has been cleared out below the pivot.

Next, we move on to the next row. The beauty of it is that we don’t have to do any new figuring! We are now working with an \(n-1 \times n-1\) system. So the next row will take \(2(n-1)^2 - 3(n-1) + 1\) operations on the left-hand side, and \(2(n-1) - 2\) on the right. And so on.

This continues until the last row where we do \(2\cdot1^2 - 3\cdot1 + 1\) operation on the left (naturally—the last row needs nothing done to it!) and \(2\cdot1 - 2 = 0\) on the right.

So how much does this add up to? We have \(\sum_{k=1}^{n} (2k^2 - 3k + 1)\) operation on the left.

Using familiar summation formulas, this is \(\frac{2n(n+1)(2n+1)}{6} - \frac{3n(n+1)}{2} + n = \frac{4n^3 - 3n^2 - n}{6}\), or less than \(2n^3/3\) operations. In fancy language, we say that elimination is \(O(n^3)\). This means that if we double the size of the matrix from \(n \times n\) to \(2n \times 2n\) we will roughly take \(2^3 = 8\) times as long to solve the problem.

Notice that the factorization \(A = LU\) is the same operation count. This is because we do all the steps to create \(U\), and the entries of \(L\) are just the multipliers which are found in the process.
On the right-hand side, we end up with \( \sum_{k=1}^{n} (2k - 2) = n(n - 1) \) steps. This is \( \mathcal{O}(n^2) \). Note that for large \( n \), this is practically irrelevant, taking only \( 1/n \)th the time, so pretty much all the time is spent doing the elimination.

**Back substitution**

OK, now how much to finish up? We now have to solve \( Ux = c \) by back substitution. The last entry takes 1 step—we divide the final entry of \( c \) by the last pivot in \( U \). Now this value is substituted into the previous equation, which reads \( u_{n-1,n-1}x_{n-1} + u_{n-1,n}x_n = c_{n-1} \). It takes one multiplication to multiply the known \( x_n \) by its coefficient, and one subtraction to subtract the result from both sides of the equation. Then one further division by the \( u \) finds \( x_{n-1} \). So this second-to-last \( x \) takes three operations.

For the third-to-last, there are now two known \( x \)'s, so each is substituted-and-subtracted (total of four steps). Then we do one more division to find \( x_{n-2} \). In other words, five steps. The next \( x \) will take seven steps, and so on, until \( x_1 \) takes \( 2n - 1 \) steps. The total here is \( n^2 \).

Note that this is the *same* total as we will need for the forward substitution used when solving \( Lc = b \) for \( c \). In fact, this last will take \( n \) fewer steps \( (n^2 - n) \) because we don’t need to do any of the divisions—we know that all the diagonal entries of \( L \) are one. So, as claimed before, the forward substitution takes exactly as many steps as the standard elimination to find \( c \). (It is, in fact, the exact same sequence of steps, just delayed until we have found the complete \( LU \).) So from now on, we’ll just do \( LU \) factorization, and then forward and back substitutions.

So each right-hand side takes \( 2n^2 - n \) steps, which is much smaller than the \( 2n^3/3 \) steps needed to do the elimination in the first place. That’s why \( LU \) is so great—we only have to do the elimination once.

**Comparison to other solution methods**

Why is elimination the method of choice for solving systems? Because other methods are much slower.

Consider finding the inverse and multiplying \( A^{-1}b \). It turns out the best way to find \( A^{-1} \) is by Gauss-Jordan elimination. Using the same kind of operation count we have just done for elimination, we can find that Gauss-Jordan will take approximately \( 8n^3/3 \) operations, or about four times with it costs to find \( LU \). And each right hand side requires we multiply \( A^{-1}b \). Well, each entry of the product is \( n \) multiplications and \( n - 1 \) additions, and there are \( n \) entries, giving \( 2n^2 - n \) total operations per each new right hand side. This is exactly the same as for the forward and back substitutions, so we gain nothing there. Thus, it is only if the inverse is needed for some other reason that we should ever bother actually calculating it.

Cramer’s rule is a disaster. Recall that Cramer’s rule is to find each entry in the solution by taking \( D_x/D \) where \( D_x \) and \( D \) are certain determinants. The familiar cofactor method of finding determinants (working down a column or across a row and finding all the determinants of the smaller matrices and combining them with the proper signs) takes \( n! \) steps. This is outrageously large compared to \( n^3 \) as soon as \( n > 5 \). It turns out that the best method for finding determinants is elimination! This is because the determinant is the product of the pivots. So *each* determinant takes \( 2n^3/3 \) steps, and there are \( n \) new determinants to compute for each new right hand side—totally about \( 2n^4/3 \) steps, instead of the \( 2n^2 \) steps for the substitutions. Yikes!

Reading: 2.6
Problems: 2.6: 1 – 4, 6, 7, 11, 12, 13, 16, 19, 20 (just factor it), 21, 25, 26