THREE MYSTERIES OF GAUSSIAN ELIMINATION

Lloyd N. Trefethen
Department of Mathematics
Massachusetts Institute of Technology

If numerical analysts understand anything, surely it must be Gaussian elimination. This is the oldest and truest of numerical algorithms. To be precise, I am speaking of Gaussian elimination with partial pivoting, the universal method for solving a dense, unstructured $n \times n$ linear system of equations $Ax = b$ on a serial computer. This algorithm has been so successful that to many of us, Gaussian elimination and $Ax = b$ are more or less synonymous. The chapter headings in the book by Golub and Van Loan [3] are typical -- along with "Orthogonalization and Least Squares Methods," "The Symmetric Eigenvalue Problem," and the rest, one finds "Gaussian Elimination," not "Linear Systems of Equations."

Yet as I have taught Gaussian elimination in my classes, I have been surprised to find that in basic ways it appears to be not understood why this algorithm occupies the position it does. The closer one looks, the more subtle and remarkable Gaussian elimination appears. The purpose of this note (which is more than usually speculative) is to sort out my own thoughts on this topic and to solicit ideas from others.

It is a pleasure to acknowledge my debt to many stimulating lectures by Gene Golub and Jim Wilkinson, whose enthusiasm for numerical linear algebra is contagious.

Complexity

My first question is the broadest, having to do with whether Gaussian elimination, or indeed any of the standard direct methods, will remain important in the long run.

QUESTION 1. Are there practical algorithms for solving $Ax = b$ in fewer than $O(n^3)$ operations?

This is a emotionally charged question among numerical analysts. Gaussian elimination and all of the other methods used in practice require $O(n^3)$ work, but as is well known, algorithms also exist that are faster -- at least in theory. First there was Strassen's $O(n^{2.81})$ algorithm in 1968 [7], and then, beginning with Pan in 1978, a succession of improvements culminating in the current best rate of $O(n^{2.495})$ due to Coppersmith and Winograd [2]. See Pan's survey paper [4] for details and for a fascinating plot of the best known exponent as a function of time. Nobody has been able to prove that $O(n^{\alpha})$ operations are required except for the obvious value $\alpha = 2$. We do know that various basic matrix problems are theoretically equivalent as regards this mysterious exponent: solving $Ax = b$, computing A^{-1}, computing $\det A$, or multiplying A by another $n \times n$ matrix B [1].

The trouble with the currently known fast algorithms is two-fold: they are unstable, and the constant factors are enormous. This is where the emotions come in. Perhaps because complexity theorists seem unconcerned with stability and constant factors -- perhaps even because they speak of "matrix multiplication" and "matrix inversion" rather than "solution of linear systems" -- many numerical analysts have regarded the question of fast algorithms with disdain. There is a widespread attitude that any method that beats $O(n^3)$ must *ipso facto* be impractical. But this view is unjustified. All we really know is the impracticality of the methods devised so far.

Conceivably, a breakthrough may lie ahead in linear algebra that will be as important as the FFT in Fourier analysis -- maybe even a simple, stable algorithm that solves $Ax = b$ in $O(n^2 \log n)$ operations. But whether or not this best of all possible situations can be achieved, it is easy to imagine that there may be important *practical* ideas in this area as yet undiscovered. Many other computational problems have seen dramatic developments in recent years, such as prime number determination and linear programming. So long as there is no clear reason for pessimism, why do numerical analysts leave the search for fast matrix algorithms to the complexity theorists?

Orthogonality

My second question is rather philosophical, but is planted at least in the solid ground of $O(n^3)$ algorithms.

QUESTION 2. Why do we use non-orthogonal methods for solving $Ax = b$, but orthogonal ones for many other linear algebra computations?

Of course this statement is oversimplified. But broadly speaking, orthogonal operations are favored for least-squares and eigenvalue calculations, and not for solving systems of equations [3,9]. Where does the difference come from?

The standard reason why we use Gaussian elimination to solve $Ax = b$ rather than a QR factorization is a factor of 2: $\frac{1}{3}n^3$ vs. $\frac{2}{3}n^3$ flops.

The standard reason why we recommend a QR factorization for least-squares computations rather than the normal equations is a matter of stability: the latter approach squares the condition number. (But again it costs only half as much, and the normal equations are very commonly used in practice.)

The standard reason why we use the QR iteration for eigenvalue computations rather than a nonorthogonal analog is again a concern for stability (at least in the nonsymmetric case) -- which seems especially important here since a sequence of transformations must be performed rather than just one.

There are two themes in these arguments:

(a) Non-orthogonal algorithms are twice as fast;

(b) Orthogonal algorithms are more stable.

As far as I know, however, no results are available to pin down to what extent either of these statements is generally valid. Regarding (a), it is tantalizing to realize that if a way were discovered to compute the QR factorization in $\frac{1}{3}n^3$ rather than $\frac{2}{3}n^3$ flops, the entire justification for Gaussian elimination would vanish. Can any theorem be established to show that this will not happen?

Assuming (a) is generally true, the weight of the decision falls on (b). But is the risk of instability truly inherent in nonorthogonal algorithms? Gaussian elimination is a striking counterexample: it is stable but not orthogonal. On the other hand its stability is much less obvious than that of alternative orthogonal algorithms. How do we know that this is not the general pattern? Might there not be factors of 2 to be saved in other areas at the price of more complicated stability analysis?

This seems already true in some instances. For example, EISPACK offers both an orthogonal and a nonorthogonal reduction of a matrix to Hessenberg form, preliminary to the determination of eigenvalues by the QR iteration [6]. There is no general consensus as to which choice is best. A similar relationship holds between Givens and "fast Givens" rotations for the element-by-element introduction of zeros [3].

Stability

My final question is completely down-to-earth, and consequently, the one whose unresolved status is the most perplexing.

QUESTION 8. Why is Gaussian elimination stable? Specifically, why is the growth of elements during elimination negligible in practice?

The standard error analysis of Gaussian elimination, due to Wilkinson, quickly reduces stability to the question of whether much growth occurs in the size of the elements as the elimination proceeds [9]. That is, if $PA = LU$, can the entries of U be much larger than those of A? In principle the answer is yes: a simple example shows that they can be 2^{n-1} times as large. But in practice, this never occurs. This empirical conclusion is so well established that we easily forget how remarkable the situation is: *nobody knows why Gaussian elimination is stable!*

Things do not happen, or fail to happen, without reasons. In this case, it is easy to guess one kind of explanation that might be appropriate -- perhaps large growth rates like 2^{n-1} correspond to unstable "modes" that are themselves somehow unstable, in the sense that computations tend to drift away from them towards stabler configurations. Whatever the reason, the average-case behavior of Gaussian elimination with partial pivoting is evidently very different from the worst-case behavior, but nothing has been proved. There is an interesting analogy here to the simplex method for linear programming, where speed is the issue rather than accuracy, and good average-case behavior has only very recently been proved by Smale [5]. For some reason the question of average-case stability of Gaussian elimination has received far less attention over the years.

Ironically, if we switch to complete pivoting, whose stability (relatively speaking) is well understood, then $\frac{1}{3}n^3$ additional comparisons are required. This brings the operation count up towards $\frac{2}{3}n^3$ flops, just as for the orthogonal methods that are understood and stable.

* * *

These three mysteries of Gaussian elimination are enough for one brief note. But the list need not end here. In the long run, for example, one wonders whether iterative methods might prove better than all of the direct ones even for arbitrary matrices -- some kind of conjugate gradient iteration with automatic, adaptive preconditioning? Many matters connected with scaling and equilibration are still poorly understood. Even the question of forward vs. backward error analysis has recently been raised again [8]. As we enter an era of parallel computing, with the rules of the game changing fast, it is intriguing that so many fundamental issues remain unsettled in the game we have been playing for decades.

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison-Wesley, Reading, Mass., 1976.

[2] D. Coppersmith and S. Winograd, On the asymptotic complexity of matrix multiplication, *SIAM J. Comput. 11* (1982), 472-492.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations,* Johns Hopkins University Press, Baltimore, Maryland 1983.

[4] V. Pan, How can we speed up matrix multiplication?, *SIAM Review 26* (1984), 393-415.

[5] S. Smale, On the average number of steps in the simplex method of linear programming, *Math. Prog. 27* (1983), 241-262.

[6] B. T. Smith, et al., *Matrix Eigensystem Routines -- EISPACK Guide,* Lect. Notes. Comp. Sci. 6, Springer, New York, 1976.

[7] V. Strassen, Gaussian elimination is not optimal, *Numer. Math. 13* (1969), 354-356.

[8] F. Stummel, Forward error analysis of Gaussian elimination I, II, *Numer. Math. 46* (1984), 365-415.

[9] J. H. Wilkinson, *The Algebraic Eigenvalue Problem,* Clarendon Press, Oxford, 1965.