Monitoring the Stability of the Triangular Factorization of a Sparse Matrix

A. M. Erisman and J. K. Reid

Received April 5, 1973

Abstract. We propose a method for bounding the numerical instability in Gaussian elimination which may result from the use of interchanges calculated from the sparsity pattern alone or calculated for another matrix having the same sparsity pattern. The computational cost of obtaining the bound is small compared with that of the elimination. Also we propose the use of a variant of iterative refinement to estimate the accuracy of the solution obtained.

We consider the use of Gaussian elimination to solve systems of linear equations

\[ Ax = b \]  

whose matrix is sparse. In particular we consider the case, usual in a large number of applications, when solutions are required to a considerable number of systems (1) having the same sparsity pattern.

It is important to include row and column interchanges in the elimination in order to preserve sparsity. Choosing these interchanges is, unfortunately, usually far more expensive than performing the elimination with the interchanges known. It is therefore desirable to use the same interchanges for as many as possible of those problems having the same sparsity pattern. To simplify notation we will assume that \( A \) is the matrix obtained after the application of these interchanges.

A major disadvantage of this scheme is that the resulting computation may be numerically unstable. A slight extension [3] of the error analysis of Wilkinson shows that, without any assumptions on the sizes of the elements of \( L \), the calculated triangular factors \( L \) and \( U \) of \( A \) satisfy the equation

\[ A + E = LU \]  

where \( E \) is a matrix of errors whose elements satisfy the inequalities

\[ |e_{ij}| \leq (3.01) \, e \, q \, m_{ij} \]  

where \( e \) is the relative accuracy of the arithmetic in use, \( q = \max_{i,j,k} \, |a_{ij}^{(k)}| \) is the largest element in any of the intermediate matrices encountered in the elimination and \( m_{ij} \) is the number of multiplications required in the calculation of \( l_{ij}(t \geq j) \) or \( u_{ij}(i \leq j) \). It must be assumed that a scaled matrix \( A \) is in use for a meaningful interpretation of inequality (3).

Interchanges are normally included to control the size of \( q \) but here we have already used interchanges for the preservation of sparsity. It is possible to calculate \( q \) directly when performing the factorization and hence recognise an un-
stable decomposition, but this direct calculation of \( q \) places a significant overhead on the factorization costs since additional computation in the innermost loop of the program is needed. We seek to reduce this cost.

Neglecting rounding errors, the elements of the \( k \)-th reduced matrix in Gaussian elimination are given by the equation

\[
a^{(k)}_{ij} = a_{ij} - \sum_{m=1}^{k} l_{im} u_{mj}, \quad k < i \leq n \quad \text{and} \quad k < j \leq n
\]  

so that by using Hölder's inequality with \( \frac{1}{p} + \frac{1}{q} = 1 \) we find the bound

\[
|a^{(k)}_{ij}| \leq |a_{ij}| + \|l_{i1}, l_{i2}, \ldots, l_{ik}\|_p \|u_{j1}, u_{j2}, \ldots, u_{jk}\|_q.
\]  

This inequality may be weakened to

\[
|a^{(k)}_{ij}| \leq \max_{i,j} |a_{ij}| + \max_i \|l_{i1}, \ldots, l_{i,i-1}\|_p \max_j \|u_{j1}, \ldots, u_{j-1,j}\|_q
\]  

which is a generalization of the bound used by Businger [1]. He took \( p = 1 \) and was assuming that partial pivoting was in use on a full matrix so the bound \( \|l_{ii}, \ldots, l_{i,i-1}\| \leq n-1 \) was available to him.

The bound (6) can be calculated very economically in each of the cases \( p = 1, 2, \infty \) since only one reference to each non-zero off-diagonal element of \( L \) and \( U \) is needed to accumulate the norms (or their squares if \( p = 2 \)). The stronger result (5) may be applied with very little extra computational effort if the norms (or their squares for \( p = 2 \)) are accumulated in \( 2n \) variables and the inequality (or the corresponding inequality for \( \|a^{(k)}_{ij}\| - |a_{ij}| \) if \( p = 2 \)) is used for elements with \( \min(i, j) = m \) just once, before the \( m \)-th stage of the elimination.

In the symmetric (real or complex) case it is desirable to replace (4) by

\[
a^{(k)}_{ij} = a_{ij} - \sum_{m=1}^{k} l_{im}^2 d_m l_{mj}, \quad k < i \leq n \quad \text{and} \quad k < j \leq n
\]  

so that both storage and computational effort may be halved. In the real and positive definite case Wilkinson [6] has shown that \( |a^{(k)}_{ij}| \leq \max_{i,j} |a_{ij}| \) and the symmetric algorithm is often thought to be applicable only in this case but our experience is that it is also very useful in the general case provided the stability is monitored. Taking \( p = 2 \) we find, by analogy with (6) the result

\[
|a^{(k)}_{ij}| \leq \max_{i,j} |a_{ij}| + \max_i \sum_{m=1}^{i-1} |l_{im}^2 d_m|.
\]  

The same bound holds for Hermitian matrices.

We may expect the bounds (5) and (6) to be quite realistic if the elements of \( L \) and \( U \) do not vary greatly in size but they may be pessimistic in cases with large variations in size. This comes from the transition from (4) to (5) and is illustrated by the following computation with \( k = 2, \ p = 1 \)

\[
2 \times 10^6 = 1 \times 10^6 + 10^6 \times 1 \leq (1 + 10^6) \times 10^6.
\]