Presented at the Symposium on High Speed Computer and Algorithm Organization, University of Illinois at Urbana-Champaign, April 13-15, 1977.

Minimal Storage Band Elimination[1]

S. C. Eisenstat,[2] M. H. Schultz,[2] A. H. Sherman[3]

Research Report #105

[1] This research was supported in part by NSF Grant MCS 76-11460, ONR Grant N00014-76-C-0277, and AFSOR Grant F49620-77-C-0037.

[2]Department of Computer Science, Yale University

[3]Department of Computer Science, University of Texas at Austin

ABSTRACT

A variation of Gaussian elimination is presented for solving band systems of linear equations on computers with limited core storage, without the use of auxiliary storage such as disk or tape. The method is based on the somewhat unusual idea of recomputing rather than saving most nonzero coefficients in the reduced triangular system, thus trading an increase in work for a decrease in storage. For a five-point problem on an n x n grid, the storage required is $\sim n^2$ versus $\sim n^3$ for band elimination, while surprisingly the work required at most doubles.

1. Introduction

Consider the system of linear equations

(S) $A\,x = b$

where the coefficient matrix A is an N x N symmetric positive definite band matrix with bandwidth m, i.e., $a_{ij} = 0$ if $|i - j| > m$ (see Figure 1). Direct methods for solving (S) are generally variations of symmetric Gaussian elimination: Form the U^tDU decomposition of A, where U is unit upper triangular and D positive diagonal, and then successively solve the triangular systems

$$U^t\,y = b, \quad D\,z = y, \quad U\,x = z.$$

The matrix U is again a band matrix with the same bandwidth as A. Variations of symmetric Gaussian elimination which take advantage of this structure to avoid storing and operating on entries outside the band are known as band elimination methods [6]. The total work (in terms of the number of multiplications and divisions) and storage required are given by

$$\theta_B(N,m) = 1/2\ Nm^2 + 7/2\ Nm - 1/3\ m^3 + O(N+m^2)$$

and

$$S_B(N,m) = Nm + O(N+m^2)$$

respectively [1].

As an example, consider the following model problem which arises from the familiar five-point finite difference discretization of the Poisson equation on the unit square with homogeneous boundary conditions. Given a uniform n x n grid in the plane (see Figure 2a), we associate a variable u_{ij} with each mesh-point (i,j) and form the system of linear equations

$$4\,u_{ij} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = f_{ij} \qquad 1 \le i,\ j \le n$$

where

$$u_{ij} = 0 \qquad i = 0,\ n+1 \quad \text{or} \quad j = 0,\ n+1$$

There are $N = n^2$ unknown variables u_{ij}, and, with the natural row-by-row ordering (see Figure 2b), the bandwidth is m = n. Thus the work and storage required to solve the linear system are $\sim 1/2\ n^4$ and $\sim n^3$ respectively.

Our model problem illustrates the behavior one often encounters in using Gaussian elimination to solve large band systems: The storage required can easily exceed the core storage available for even moderately large N and m, even though the problem and solution (i.e., A, b, and x) CAN be represented in core. Thus, although we could store the nonzero coefficients, right hand side, and solution for our model problem in $\sim 5n^2$ locations, the factorization would require an additional $\sim n^3$ locations. More generally, although we could store x in N words of storage, we would still require an additional $\sim$ Nm words of storage for D and the band of U, even if we could recompute the nonzero entries of A and b as easily as store them.

In this paper we discuss several variations of band elimination which can solve (S) with minimal core storage. The methods are based on the following assumptions:

(1) The nonzero entries of A and b are inexpensive to generate on demand (e.g., A is sparse and can be represented more compactly than in band form, or A must be preserved for subsequent computations).

(2) There is enough core storage for the solution vector x, plus an additional $\gamma(N+m^2)$ words of working storage (for some small constant γ), but not enough to store the band of U.

We count only the working storage required to solve (S) in stating the storage requirements of such methods. All other storage (i.e., storage for A, b, and x) is associated with the linear system rather than the method of solution and is ignored.

In section 2, we review the standard approach to this problem, the use of auxiliary storage, and indicate some of the disadvantages of auxiliary storage band elimination. In section 3, we introduce minimal storage methods as a corollary of the somewhat unusual idea of recomputing rather than saving most nonzero entries in the factorization,[1] the effect being to trade a logarithmic increase in work for a factor of N/2m decrease in storage. Finally, in Section 4, we consider the special case of the model problem, and show that the work at most doubles while storage is reduced from $\sim n^3$ to $\sim n^2$.

---

[1] The same approach can be applied to sparse elimination; see [2].

2. Auxiliary Storage Band Elimination

A standard approach to problems in which the storage required exceeds the amount of core storage available is to use auxiliary storage such as disk or tape. This approach works extremely well for band elimination [5], as can be seen by examining the process more closely.

In the classic view of Gaussian elimination, we use the k^{th} equation to eliminate the k^{th} variable from the remaining N-k equations for k = 1,...,N, and then solve the resulting triangular system. In terms of the factorization model, this corresponds to solving

$$U^t y = b, \qquad D z = y$$

as A is factored, and then solving

$$U x = z$$

when the factorization is complete.

A program fragment for symmetric band elimination using a scratch array M appears as Figure 3a. Note that only those locations M[i,j] for which $k \leq i \leq j \leq k+m$ are referenced in eliminating the k^{th} variable. These locations form an (m+1)x(m+1) triangular window on the upper band of M. All the previous rows contain entries of U and will not be needed until back-solution; all subsequent columns contain entries of A and have not yet been used in the elimination process; and the window advances as each variable is eliminated (see Figure 3b).

This suggests an auxiliary storage band elimination algorithm. We use core storage as an (m+1)x(m+1) triangular window on the elimination process. After each variable is eliminated, the window is shifted; the row of U left behind is written to auxiliary storage; and the last column of the window is initialized to the next column of A. During back-solution, we retrieve the rows of U in reverse order and solve for the corresponding component of the solution. Since we do exactly the same operations as in band elimination,

$$\theta_{AS}(N,m) \cong 1/2\ Nm^2,$$

yet the core storage required (other than that for A, b, and x which we have agreed to ignore) is just that for the window, namely

$$S_{AS}(N,m) = 1/2\ (m+1)(m+2).$$

An obvious refinement of this algorithm is to save as much of the factorization in core as possible, but, when core storage is exhausted, to make room by moving those rows of U not needed for the next stage of elimination (i.e., not in the window) to auxiliary storage. In a certain sense, in-core band elimination on computers with large virtual memory has the same effect: When no real memory is available, the operating system moves the least recently used pages (i.e., rows of U) out to the swapping device (i.e., auxiliary storage), and then retrieves them as they are referenced during back-solution.

While auxiliary storage band elimination can be used to solve large band systems of linear equations on computers with limited core storage, it does have certain disadvantages. First, a large amount of auxiliary storage is required to save the factorization. Second, the added cost of transferring the factorization to and from auxiliary storage can not be neglected. While in principle this input/output can partially overlap the numerical computation, in practice the degree of such overlap is highly dependent on the machine, auxiliary storage device, and operating system in use. Third, retrieving the rows of U in reverse order requires the ability to read backward or random access auxiliary storage, an operation which is again highly dependent on the program environment.

3. Minimal Storage Band Elimination

The disadvantages associated with auxiliary storage band elimination are inherent in the use of auxiliary storage. In this section, we introduce a minimal storage band elimination algorithm for solving band systems of linear equations with limited core storage and NO auxiliary storage. Rather than trade input/output and auxiliary storage for a reduction in core storage, we trade additional arithmetic operations instead.[2] The basic idea behind the algorithm is quite simple. We discard most nonzero coefficients in the reduced triangular system as they are computed and regenerate them during back-solution; only enough information is retained to solve for a subset of the unknowns at each stage, thus reducing the size of the problem to be solved.

Observe that, in solving (S) using auxiliary storage band elimination, after the $(N-m)^{th}$ variable has been eliminated, the coefficients and right hand side of the resulting m x m system remain in core. Thus we can solve for the last m components of the solution without referencing auxiliary storage. Moreover, we can use these values to reduce the size of the problem to be solved. Writing (S) as a block system conforming to the N-m unknown variables x_1 and the m known variables x_2,

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{12}^t & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

we see that the remaining unknowns x_1 satisfy the reduced system

$$A_{11}x_1 = b_1 - A_{12}x_2 \equiv \tilde{b}_1.$$

This suggests the following method for solving band systems with limited core storage and no auxiliary storage:

(1) Use band elimination to eliminate the first N-m variables, discarding the rows of U as they are computed (i.e., we store only the window).

(2) Solve the resulting dense positive definite system of m equations in the last m variables.

---

2 Advances in semiconductor technology have made the central processing unit (as well as attached processors such as array processors) relatively inexpensive compared to auxiliary storage devices. Thus, such an exchange is not unreasonable.

(3) Substitute these values into the original system to obtain a new system of N-m equations in the remaining N-m unknowns.

(4) Solve the new system by the same algorithm.[3]

Clearly the storage required is the same as that for auxiliary storage band elimination, i.e.

$$S(N,m) \;=\; 1/2\;(m+1)(m+2);$$

but what is the increased cost? Letting $\theta(n,m)$ denote the work required, we have[4]

$$\theta(N,m) \;\cong\; \theta_B(N,m) \;+\; \theta(N-m,m)$$

$$\cong\; 1/2\;Nm^2 \;+\; \theta(N-m,m)\;.$$

Therefore,

$$\theta(N,m) \;\cong\; 1/4\;N^2m,$$

i.e., the work has increased by a factor of N/2m. For our model problem, we have reduced the storage required from $\sim n^3$ to $\sim 1/2\;n^2$, but the work has increased from $\sim 1/2\;n^4$ to $\sim 1/4\;n^5$. Fortunately, we can do much better.

A basic technique for speeding up algorithms is divide-and-conquer. Rather than reduce a problem to a slightly smaller subproblem at each stage, it is often more efficient to reduce it to TWO subproblems each of at most HALF the size. Thus, suppose we could solve for the middle m components of the solution. If we write (S) as a block system conforming to the first $\sim (N-m)/2$ variables x_1, the middle m variables x_2, and the last $\sim (N-m)/2$ variables x_3,

---

[3] Note that the coefficient matrix in the new system, being a principal submatrix of A, is again a symmetric positive definite band matrix with bandwidth (at most) m; thus the procedure can be applied to the new system.

[4] For ease of exposition, we have assumed that the bandwidth does not decrease and that only m components are solved for at each stage; otherwise the work would be somewhat smaller.

$$\begin{pmatrix} A_{11} & A_{12} & 0 \\ A_{12}^t & A_{22} & A_{23} \\ 0 & A_{23}^t & A_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix},$$

then the remaining unknowns x_1 and x_3 satisfy two INDEPENDENT $\tilde{}(N-m)/2 \times \tilde{}(N-m)/2$ linear systems,

$$A_{11} x_1 = b_1 - A_{12} x_2 \equiv \tilde{b}_1$$

$$A_{33} x_3 = b_3 - A_{32}^t x_2 \equiv \tilde{b}_3.$$

Since the coefficient matrices of the two subproblems are again principal submatrices of A, they are symmetric positive definite band matrices with bandwidth (at most) m. Thus the algorithm can be applied recursively to each of the subproblems. However, to use this technique, we need a method for solving for the middle m components of the solution.

Recall that symmetric Gaussian elimination can be viewed as an elimination process. That is, each equation is used to eliminate the corresponding variable from the remaining equations, and the resulting triangular system is solved for the unknown variables. For band elimination, we observed that, when we eliminate the variables in forward order, no nonzero coefficients are created outside the band. Thus we need only store and operate on coefficients within the band. Of course, the same is true if we eliminate the variables in reverse order, since reversing the order of the variables and equations preserves the band structure. Moreover, since A is positive definite, any order of elimination is numerically stable [7].

This suggests another minimal storage band elimination method:

(1) Eliminate the first ~ (N-m)/2 unknowns in forward order, using a triangular window and discarding the rows of U.

(2) Eliminate the last ~ (N-m)/2 unknowns in reverse order, again using a triangular window.[5]

(3) Solve the resulting dense positive definite system of m equations in the middle m unknowns.

---

[5] Evans and Hatzopoulos [3] have used two-sided band elimination to solve centro-symmetric ($a_{ij} = a_{n-i+1,n-j+1}$) band systems of linear equations.

(4) Use these values to split the problem into two independent subproblems.

(5) Apply the algorithm recursively to solve each subproblem.

The storage required is approximately twice that of our earlier method, since we must maintain the contents of both windows; in fact, precisely[6]

$$S_{MS}(N,m) = (m+1)^2$$

locations are required. Let $\theta_{MS}(N,m)$ denote the work required to solve a system of N equations with bandwidth m using this algorithm. The work to solve for the middle m unknowns is exactly the same as before so that[7]

$$\begin{aligned}\theta_{MS}(N,m) &\cong \theta_B(N,m) + 2\,\theta_{MS}((N-m)/2,m)\\ &\cong 1/2\,N\,m^2 + 2\,\theta_{MS}(N/2,m)\quad .\end{aligned}$$

Thus

$$\theta_{MS}(N,m) \cong 1/2\,Nm^2\,\log_2(2N/m).$$

While minimal storage band elimination avoids the problems associated with the use of auxiliary storage, it does have certain disadvantages. First, the work required has increased logarithmically, although this can be improved somewhat by reordering the subproblems at each stage to minimize bandwidth (see Section 4). Second, the method has no memory: We solve for one right hand side but do not save any of the information needed to solve for additional right hand sides (as in auxiliary storage band elimination). Third, the method is somewhat more complex than in-core or auxiliary storage band elimination.

----

6 Storing two complete windows would require (m+1)(m+2) locations. However, when reverse elimination (Step (2)) begins, the last column of the forward window contains an unmodified column of A and need not be stored.

7 Again, for ease of exposition, we have assumed that the bandwidth does not decrease.

4. Minimal Storage Band Elimination for the Model Problem

In deriving the operation count for minimal storage band elimination, we did not make any additional assumptions about the induced subproblems. We merely observed that the coefficient matrices, being principal submatrices of A, were again symmetric and positive definite with bandwidth (at most) m. As a consequence, the decrease in storage engendered a logarithmic increase in work. However, if the matrix A has more structure, one would expect that the bandwidths of some subproblems could be reduced by reordering the variables and equations, thus reducing the total work required. The savings can be quite significant. We show in this section that the number of arithmetic operations needed to solve our model problem is approximately 5/3 that for band elimination.[8]

Consider minimal storage band elimination applied to a five-point problem on an n x n grid. During the first partial solution step, we solve for the variables on a horizontal grid line[9] which divides the grid into two ~n/2 x n subgrids (see Figure 4a). Given these values, the problem splits into independent five-point problems on each of the two subgrids. The optimum ordering for each subproblem (in terms of minimizing the bandwidth) is the column-by-column ordering for which the bandwidth is ~n/2 [4]. During the second partial solution step (as applied to the optimally ordered subproblems), we solve for the variables on vertical grid lines[9] which further subdivide the grid (see Figure 4b). Each resulting subproblem is a five-point problem on an ~n/2 x ~n/2 grid. Thus we have reduced our problem to four structurally similar problems of approximately one-fourth the size which can now be solved in the same manner.[10]

---

[8] Similar results are easily obtained for more general finite difference and finite element approximations to self-adjoint elliptic partial differential equations. The key property is that the subproblems generated have a sufficiently regular structure that one can analyze the effect of reordering to minimize bandwidth.

[9] When n is even, the middle m variables do not lie on a single grid-line; however, we could equally well solve for the m variables on either of the two central grid-lines.

[10] It is interesting to note that the order in which we solve for the variables is essentially the reverse of the nested dissection ordering proposed by George [4] for sparse elimination.

Let $\theta_{MS}(p)$ denote the number of arithmetic operations required to solve a five-point problem on a p x p grid using optimally ordered minimal storage band elimination. From the preceding discussion, we have

$$\begin{aligned}\theta_{MS}(p) &\cong \theta_B(p^2,p) + 2\,\theta_B(p^2/2,p/2) + 4\,\theta_{MS}(p/2)\\ &\cong 1/2\,(p^2)(p)^2 + 2(1/2)(p^2/2)(p/2)^2\\ &\quad + 4\,\theta_{MS}(p/2) + O(p^3)\\ &\cong 4\,\theta_{MS}(p/2) + 5/8\,p^4 + O(p^3).\end{aligned}$$

Therefore

$$\theta_{MS}(n) \cong 5/6\,n^4 + O(n^3)$$

so that the total work is approximately 5/3 that for band elimination. The total storage is just

$$S_{MS}(n) = (n+1)^2$$

versus $\sim n^3$ for band elimination.

During the execution of this algorithm, we eventually reach a value of p sufficiently small that the entire factorization of each p x p subproblem can be kept in core. In particular, if

$$p \le n^{2/3},$$

then the storage required for band elimination satisfies

$$S_B(p^2,p) \cong (p^2)(p) = p^3 \le (n^{2/3})^3 = n^2,$$

whereas we have assumed that the amount of storage available is at least $\sim n^2$. Thus we could switch to band elimination at this point. The switch would have very little effect on the total number of arithmetic operations since most of the work is done in the first few stages. None-the-less, the reduced book-keeping makes it a good idea in practice.

Figure 1. An N x N symmetric band matrix with bandwidth m (N = 9, m = 4)

$$\begin{bmatrix}
X & X & X & X & & & & & \\
X & X & X & X & X & & & & \\
X & X & X & X & X & X & & & \\
X & X & X & X & X & X & X & & \\
 & X & X & X & X & X & X & X & \\
 & & X & X & X & X & X & X & X \\
 & & & X & X & X & X & X & X \\
 & & & & X & X & X & X & X \\
 & & & & & X & X & X & X
\end{bmatrix}$$

Figure 2a. An nxn grid.

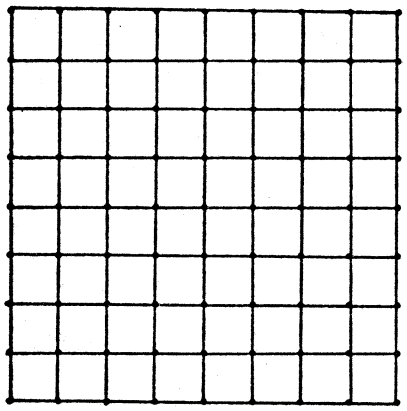

Figure 2b. The natural row-by-row ordering of an nxn grid.

```
*  *  *  *  *  *  *  *  *
*  1  2  3  4  5  6  7  *
*  8  9 10 11 12 13 14  *
* 15 16 17 18 19 20 21  *
* 22 23 24 25 26 27 28  *
* 29 30 31 32 33 34 35  *
* 36 37 38 39 40 41 42  *
* 43 44 45 46 47 48 49  *
*  *  *  *  *  *  *  *  *
```

Figure 2c. The coefficient matrix for the model problem on an nxn grid.

$$A = \begin{bmatrix} T & -I & & & & \\ -I & T & -I & & & \\ & -I & T & -I & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & -I & T & -I \\ & & & & -I & T \end{bmatrix}_{nxn} \qquad T = \begin{bmatrix} 4 & -1 & & & & \\ -1 & 4 & -1 & & & \\ & -1 & 4 & -1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & -1 & 4 & -1 \\ & & & & -1 & 4 \end{bmatrix}_{nxn}$$

Fig. 3a. Symmetric band elimination.

```
FOR k = 1 UNTIL N DO
   {imax = MIN (k+m, N);
    FOR i = k UNTIL imax DO
       M[k,i] = A[k,i];
    x[k] = b[k]};

FOR k = 1 UNTIL N DO
   {imax = MIN (k+m, N);
    FOR i = k+1 UNTIL imax DO
       {uik = M[k,i] / M[k,k];
        FOR j = i UNTIL imax DO
           M[i,j] = M[i,j] - uik * M[k,j];
        x[i] = x[i] - uik * x[k];
        M[k,i] = uik};
    x[k] = x[k] / M[k,k]};

FOR k = N STEP -1 UNTIL 1 DO
   {imax = MIN (k+m, N);
    FOR i = k+1 UNTIL imax DO
       x[k] = x[k] - M[k,i] * x[i]};
```

Figure 3b. The window on the band of A advances at each step.

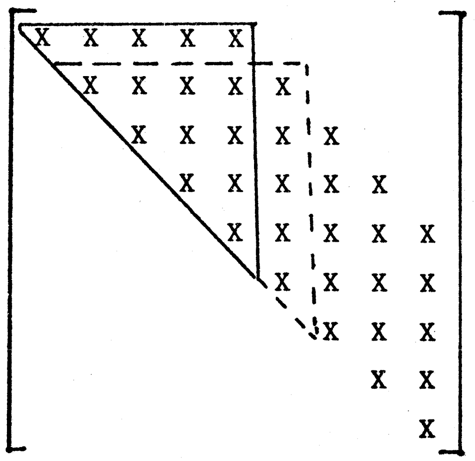

Figure 4a. The variables on a horizontal grid line subdivide the grid.

```
o o o o o o o                 o o o o o o o
o o o o o o o                 o o o o o o o
o o o o o o o                 o o o o o o o
x x x x x x x      ----->
o o o o o o o                 o o o o o o o
o o o o o o o                 o o o o o o o
o o o o o o o                 o o o o o o o
```

Figure 4b. The variables on vertical grid lines further subdivide the grid.

```
o o o x o o o                 o o o   o o o
o o o x o o o                 o o o   o o o
o o o x o o o                 o o o   o o o
                   ----->
o o o x o o o                 o o o   o o o
o o o x o o o                 o o o   o o o
o o o x o o o                 o o o   o o o
```

REFERENCES

1. Bunch, J. R., Analysis of sparse elimination, SIAM Journal on Numerical Analysis 11:847-873, 1974.

2. Eisenstat, S. C., M. H. Schultz, and A. H. Sherman, Minimal storage sparse elimination. To appear.

3. Evans, D. J., and M. Hatzopoulos, The solution of certain banded systems of linear equations using the folding algorithm, Computer Journal 19:184-187, 1976.

4. George, J. A., Nested dissection of a regular finite element mesh, SIAM Journal on Numerical Analysis 10:345-363, 1973.

5. Jennings, A., and A. D. Tuff, A direct method for the solution of large sparse symmetric simultaneous equations. In J. K. Reid, Editor, Large Sparse Sets of Linear Equations, Academic Press, 1971.

6. Martin, R. S., and J. H. Wilkinson, Symmetric decomposition of positive definite band matrices, Numerische Mathematik 7:355-361, 1965.

7. Wilkinson, J. H., The Algebraic Eigenvalue Problem, Clarendon Press, 1965.