STABILITY OF THE PARTITIONED INVERSE METHOD FOR PARALLEL SOLUTION OF SPARSE TRIANGULAR SYSTEMS*

NICHOLAS J. HIGHAM† AND ALEX POTHEN‡

Abstract. Several authors have recently considered a parallel method for solving sparse triangular systems with many right-hand sides. The method employs a partition into sparse factors of the product form of the inverse of the coefficient matrix. It is shown here that while the method can be unstable, stability is guaranteed if a certain scalar that depends on the matrix and the partition is small, and that this scalar is small when the matrix is well-conditioned. Moreover, when the partition is chosen so that the factors have the same sparsity structure as the coefficient matrix, the backward error matrix can be taken to be sparse.

Key words. sparse matrix, triangular system, substitution algorithm, parallel algorithm, rounding error analysis, matrix inverse.

AMS subject classifications. primary 65F05, 65F50, 65G05

1. Introduction. The method of choice for solving triangular systems on a serial computer is the substitution algorithm. Several approaches have been suggested for parallel solution. Implementations of substitution for distributed memory architectures are described by Heath and Romine [12] and Li and Coleman [15], and a short survey of this work is given by Gallivan, Plemmons and Sameh [9, Sec. 3.5.2] (see also [11, Sec. 6.4.4]). Implementations of substitution for sparse matrices on shared memory architectures are investigated by Rothberg and Gupta [20]. Algorithms that are not based on substitution are surveyed by Gallivan et al. [9, Sec. 3.5], Heller [13] and Ortega and Voigt [16]. A new method has been developed recently for the parallel solution of sparse triangular systems with many right-hand sides when these vectors are not necessarily available at the same time [1], [2], [3], [8]. The method involves representing the inverse of the coefficient matrix as a product of sparse factors, and can be explained as follows.

If $L \in \mathbb{R}^{n\times n}$ is lower triangular we can write $L = L_1 L_2 \dots L_n$, where L_k differs from the identity matrix only in the kth column:

$$L_k = \begin{bmatrix} I_{k-1} & & & & \\ & l_{kk} & & & \\ & l_{k+1,k} & 1 & & \\ & \vdots & & \ddots & \\ & l_{nk} & & & 1 \end{bmatrix}. \tag{1.1}$$

The factorization of L can be partitioned

$$L = G_1 G_2 \dots G_m, \tag{1.2}$$

*In *SIAM J. Sci. Comput.*, 15(1):139–148, Jan. 1994

†Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (na.nhigham@na-net.ornl.gov). The work of this author was supported by a Nuffield Science Research Fellowship, by Science and Engineering Research Council grant GR/H52139, and by the EEC Esprit Basic Research Action Programme, Project 6634 (APPARC).

‡Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada (apothen@narnia.uwaterloo.ca, na.pothen@na-net.ornl.gov). This author was supported by NSF grant CCR-9024954 and by U. S. Department of Energy grant DE-FG02-91ER25095 at the Pennsylvania State University, and by the Canadian Natural Sciences and Engineering Research Council under grant OGP0008111 at the University of Waterloo.

where $1 \le m \le n$ and

$$G_k = L_{i_k} L_{i_k+1} \dots L_{i_{k+1}-1}, \qquad 1 = i_1 < i_2 < \cdots < i_{m+1} = n+1. \tag{1.3}$$

Note that G_k is the lower triangular matrix equal to the identity except for columns $i_k, \dots, i_{k+1}-1$, which equal the corresponding columns of $L_{i_k}, \dots, L_{i_{k+1}-1}$, respectively. Defining $\overline{G}_k = G_k(:, i_k\colon i_{k+1}-1)$ (using the colon notation from [11]) we have the relation

$$L = G_1 G_2 \dots G_m = [\overline{G}_1,\ \overline{G}_2,\ \dots,\ \overline{G}_m], \tag{1.4}$$

which we will use later. Equation (1.2) yields the partitioned, product-form representation

$$L^{-1} = H_m H_{m-1} \dots H_1, \qquad H_k = G_k^{-1}. \tag{1.5}$$

For a sparse matrix L, the idea behind the "partitioned inverse method" is to choose the partition (1.2) so that (1.5) represents L^{-1} as a short product of sparse factors. Then $Lx = b$ is solved by forming

$$x = H_m H_{m-1} \dots H_1 b, \tag{1.6}$$

and the advantage is that x can be computed in m serial steps of parallel matrix-vector multiplication. Thus on a massively parallel SIMD computer such as the Connection Machine CM-2, only m communication steps involving the router are necessary in the algorithm. The scalar multiplications in each product $x_k = H_k x_{k-1}$ (where $x_0 = b$) can be done concurrently in time proportional to ν, where ν is the maximum number of elements of H_k assigned to a processor, and the additions can be done in logarithmic time [1]. The two extreme cases are $m = n$, which gives a modified version of forward substitution (or forward substitution itself if L has unit diagonal), and $m = 1$, for which the method forms $x = L^{-1} \times b$. For a sparse matrix L we would not take $m = 1$, because $H_1 = L^{-1}$ is usually much denser than L [7, Sec. 12.6]. Rather, we would like to minimize m subject to the condition that each factor H_k can be stored in the same space as G_k, since m is the number of serial steps in the parallel evaluation of x. Since we are assuming that many right-hand sides are to be processed, we can afford to spend some computational effort in constructing the partition (1.2).

Algorithms for finding a *best no-fill partition* (1.2) are described in [1, 2, 3]; such a partition has the smallest possible number of factors (the minimum value of m) subject to the requirement that each G_k is invertible in place. A matrix X is *invertible in place* if $(X^{-1})_{ij} = 0$ whenever $x_{ij} = 0$, for any assignment of (nonzero) numerical values to the nonzeros in X. Note that L_k in (1.1) is invertible in place, so a partition with $m = n$ is always a no-fill partition. When L is sparse, a best no-fill partition could have $m \ll n$. Partitions that incur some fill-in have also been investigated [3].

Algorithms are also given in [1, 2] for finding a *best reordered partition*: this is a no-fill partition with the fewest factors over all lower triangular matrices PLP^T, where P is a permutation matrix. Let $F = L + L^T$ denote the *filled matrix* corresponding to a Cholesky factor. It is well-known that if L is the Cholesky factor of a symmetric positive definite matrix A whose nonzero elements are algebraically independent, then the adjacency graph of F is chordal. By exploiting chordality, very efficient algorithms for computing best reordered partitions in time and space linear in the order of the matrix (rather than the number of nonzeros) can be designed for a Cholesky factor

L [1, 19]. Furthermore, algorithms for finding a partition with the fewest factors over all permutations P such that the permuted matrix PFP^T has the same structure as the filled matrix F have also been designed [17], [18]. Note that in this case, the permutation may change the structure of L, and hence the permutation P has to be applied to A *before* it is factored.

The numerical stability of the partitioned inverse method has not been studied in previous work, either theoretically or in numerical experiments. The numerical stability is clearly questionable because when $m = 1$ (which gives the best no-fill partition for a dense matrix) the method computes $x = L^{-1} \times b$, and a numerical example in [6, Sec. 4] shows that this evaluation need not be backward stable. To answer the question of stability we have done an error analysis of the partitioned inverse method; this analysis is presented in §2. In §3 we describe some numerical experiments that illustrate the analysis and confirm the possible numerical instability of the method.

Our main findings are as follows.

(1) In general, the partitioned inverse method does not satisfy the componentwise backward and forward error bounds enjoyed by the substitution algorithm (namely, (2.1) and (2.2)).

(2) Normwise stability depends on a quantity ρ, defined in (2.13), which is a function of the matrix L and the partition, and which can be arbitrarily large. Specifically, the computed solution $\widehat{x}$ to $Lx = b$ satisfies $(L + \Delta L)\widehat{x} = b$, where $\|\Delta L\|_\infty$ is bounded in (2.12); the relative error $\|x - \widehat{x}\|_\infty / \|x\|_\infty$ is bounded in (2.19). If ρ is of order 1, which is guaranteed if L is well-conditioned, the partitioned inverse method is both normwise backward stable and normwise forward stable.

(3) If L is sparse and each G_k is invertible in place (as is guaranteed by a best no-fill or best reordered partition), then the backward error matrix ΔL mentioned in (2) can be taken to have the same sparsity structure as L.

Another way to summarize the stability of the partitioned inverse method is to say that the method is only conditionally stable, with the backward error dependent on the condition number of L. The partitioned inverse method method therefore provides another example, to add to those discussed by Demmel [4], of how parallelism can conflict with stability.

In future work we intend to examine how particular sparsity structures and other special properties of L affect the stability of the partitioned inverse method.

2. Error analysis. In this section we give an error analysis of the partitioned inverse method for solving $Lx = b$. To keep the analysis general we will not make any assumptions about sparsity. As our model of floating point arithmetic we take

$$\begin{aligned} fl(x \pm y) &= x(1+\alpha) \pm y(1+\beta), \qquad |\alpha|, |\beta| \le u, \\ fl(x \operatorname{op} y) &= (x \operatorname{op} y)(1+\delta), \qquad |\delta| \le u, \quad \operatorname{op} = *, /, \end{aligned}$$

where u is the unit roundoff. This model admits machines that lack a guard digit in addition and subtraction. We place a hat over a variable to indicate a computed quantity.

For later comparison we summarise what can be said about the substitution algorithm. The computed $\widehat{x}$ satisfies (see, for example [22, p. 150], or [14])

$$(L + \Delta L)\widehat{x} = b, \qquad |\Delta L| \le \big((n+1)u + O(u^2)\big)|L|. \tag{2.1}$$

(Absolute values and inequalities are interpreted componentwise for matrices.) This result shows that there is a componentwise tiny backward error matrix ΔL that has

the same sparsity structure as L. From (2.1) it is easy to obtain the forward error bound

$$\frac{\|x-\widehat{x}\|_\infty}{\|x\|_\infty} \le (n+1)u\,\mathrm{cond}(L,x) + O(u^2), \tag{2.2}$$

which contains the Bauer–Skeel condition number

$$\mathrm{cond}(L,x) = \frac{\| \,|L^{-1}||L||x|\,\|_\infty}{\|x\|_\infty}.$$

This bound may be weakened to

$$\frac{\|x-\widehat{x}\|_\infty}{\|x\|_\infty} \le (n+1)u\kappa_\infty(L) + O(u^2), \tag{2.3}$$

where $\kappa_\infty(L) = \|L\|_\infty\|L^{-1}\|_\infty$. Our aim is to see how close the partitioned inverse method comes to achieving the ideal bounds (2.1) and (2.2). Note that if L is sparse, the constant $n+1$ in (2.1)–(2.3) can be replaced by $p+1$, where p is the maximum number of nonzeros per row over all the rows of L. Similarly, the constants c_k that we define below to equal the partition widths can be redefined to take account of sparsity.

First, we consider the computation of the factors $H_k = G_k^{-1}$ of L^{-1} (from L). Because of its special structure, G_k is formed without error, and we assume that G_k^{-1} is computed by one of the several stable methods described by Du Croz and Higham in [6] (for example, its columns may be computed one at a time by forward substitution). For each of these methods applied to G_k, precisely one of the following two residual bounds holds, depending on the method:

$$|\widehat{H}_k G_k - I| \le \big(c_k u + O(u^2)\big)|\widehat{H}_k||G_k|, \tag{2.4}$$

$$|G_k \widehat{H}_k - I| \le \big(c_k u + O(u^2)\big)|G_k||\widehat{H}_k|, \tag{2.5}$$

where $c_k = i_{k+1} - i_k + 1$. Each residual bound implies the forward error bound

$$|\widehat{H}_k - G_k^{-1}| \le c_k u|H_k||G_k||H_k| + O(u^2). \tag{2.6}$$

(Since we are working to first order, $\widehat{H}_k$ and H_k are interchangeable in all the bounds.)

Applying standard error analysis of matrix-vector products [11, p. 66] to (1.6) we obtain

$$\widehat{x} = (\widehat{H}_m + \Delta_m)(\widehat{H}_{m-1} + \Delta_{m-1})\ldots(\widehat{H}_1 + \Delta_1)b, \tag{2.7}$$

where

$$|\Delta_k| \le \big(c_k u + O(u^2)\big)|\widehat{H}_k|. \tag{2.8}$$

If the inner products that occur in the matrix vector products are evaluated using the fan-in algorithm for summation, then the constant c_k in (2.8) can be replaced by $\log_2 c_k$. We can rewrite (2.7) as $(L+\Delta L)\widehat{x} = b$, where

$$L + \Delta L = (H_1 + E_1)^{-1}(H_2 + E_2)^{-1}\ldots(H_m + E_m)^{-1}, \tag{2.9}$$

with

$$E_k = \Delta_k + (\widehat{H}_k - H_k). \tag{2.10}$$

Now we consider the sparsity of ΔL. First, we note that if H_k is computed by forward substitution then, by (2.1), its jth column $\widehat{h}_j$ satisfies $(G_k + F_k^{(j)})\widehat{h}_j = e_j$, where e_j is the jth unit vector and $|F_k^{(j)}| \le \big((n+1)u + O(u^2)\big)|G_k|$, so that $F_k^{(j)}$ has the same sparsity structure as G_k. It follows that if G_k is invertible in place then $\widehat{H}_k$ (whose jth column is that of $(G_k + F_k^{(j)})^{-1}$) has the same sparsity structure as G_k. The same is true for any of the stable methods in [6], because, as explained there, these methods all incur essentially the same rounding errors. Next, we observe that by (2.8), Δ_k has the same sparsity structure as $\widehat{H}_k$, and therefore if G_k is invertible in place then $(H_k + E_k)^{-1} = (\widehat{H}_k + \Delta_k)^{-1}$ has the same sparsity structure as G_k. From (2.9) and the structural relation $L = G_1 G_2 \dots G_m$, we conclude that *if each G_k is invertible in place then the backward error matrix ΔL has the same sparsity structure as L.* It remains to bound ΔL.

From (2.9) we have

$$\Delta L = -\sum_{k=1}^{m} H_1^{-1} \dots H_{k-1}^{-1} \cdot H_k^{-1} E_k H_k^{-1} \cdot H_{k+1}^{-1} \dots H_m^{-1} + O(u^2),$$

so that

$$|\Delta L| \le \sum_{k=1}^{m} |H_1^{-1} \dots H_{k-1}^{-1}| \cdot |H_k^{-1} E_k H_k^{-1}| \cdot |H_{k+1}^{-1} \dots H_m^{-1}| + O(u^2).$$

If (2.4) holds, then, from (2.8) and (2.10),

$$\begin{aligned} |H_k^{-1} E_k H_k^{-1}| &= |H_k^{-1}\Delta_k H_k^{-1} + H_k^{-1}(\widehat{H}_k H_k^{-1} - I)| \\ &\le c_k u |H_k^{-1}||H_k||H_k^{-1}| + c_k u |H_k^{-1}||H_k||G_k| + O(u^2) \\ &= 2c_k u |G_k||H_k||G_k| + O(u^2), \end{aligned}$$

and precisely the same bound is obtained if we use (2.5).

Define $d_n = 2\max_k c_k$. To obtain a convenient bound for ΔL we use (1.4), together with the observation that since $|G_k||H_k||G_k|$ differs from the identity only in columns $i_k, \dots, i_{k+1} - 1$, it can be treated like $|G_k|$ when we invoke (1.4). We have

$$\begin{aligned} |\Delta L| &\le d_n u \sum_{k=1}^{m} |G_1| \dots |G_{k-1}| \cdot |G_k||H_k||G_k| \cdot |G_{k+1}| \dots |G_m| + O(u^2) \\ &\le d_n u \sum_{k=1}^{m} \big[|\overline{G}_1|,\ \dots,\ |\overline{G}_{k-1}|,\ \overline{|G_k||H_k||G_k|},\ |\overline{G}_{k+1}|,\ \dots,\ |\overline{G}_m|\big] + O(u^2) \\ &\le d_n u \big((m-1)|L| + \sum_{k=1}^{m} [0,\ \dots,\ 0,\ \overline{|G_k||H_k||G_k|},\ 0,\ \dots,\ 0]\big) + O(u^2) \\ &= d_n u \big((m-1)|L| + \sum_{k=1}^{m} |G_k||H_k||G_k| - (m-1)I\big) + O(u^2). \end{aligned} \tag{2.11}$$

This bound is not of the form (2.1) that holds for substitution, because of the summation term. If $m = 1$, the bound is $|\Delta L| \le 2(n+1)u|L||L^{-1}||L| + O(u^2)$. When $m = n$, the relation $|L_k||L_k^{-1}||L_k| \le 3|L_k|$ allows us to simplify the bound to $|\Delta L| \le 4(n+2)u|L| + O(u^2)$, which is of the same form as in (2.1).

Taking norms in (2.11) we obtain

$$\|\Delta L\|_\infty \le d_n u(m-1+\rho)\|L\|_\infty + O(u^2), \tag{2.12}$$

where

$$\rho = \frac{\left\|\sum_{k=1}^m |G_k||G_k^{-1}||G_k| - (m-1)I\right\|_\infty}{\|L\|_\infty}. \tag{2.13}$$

The scalar $\rho \ge 1$ might be loosely described as a growth factor for the partitioned inverse method, although it is not related to the growth factor in Gaussian elimination. For any $m < n$, ρ can be arbitrarily large, but for $m = n$ it is easy to show that $\rho \le 3$. Under scaling of the system, ρ behaves as follows: if $D_1 = \mathrm{diag}(d_i)$ and $D_2 = \mathrm{diag}(e_i)$ are nonnegative diagonal matrices and we scale $Lx = b \to (D_1 L D_2)\cdot(D_2^{-1}x) = D_1 b$, then

$$\rho \to \overline{\rho} = \frac{\left\|\sum_{k=1}^m D_1^{(k)}|G_k||G_k^{-1}||G_k|D_2^{(k)} - (m-1)I\right\|_\infty}{\|D_1 L D_2\|_\infty},$$

where $D_1^{(k)} = \mathrm{diag}(1,\dots,1,d_{i_k},\dots,d_n)$ and $D_2^{(k)} = \mathrm{diag}(1,\dots,1,e_{i_k},\dots,e_{i_{k+1}-1},1,\dots,1)$ (recall that i_k is defined in (1.3)). This expression suggests that ρ is fairly insensitive to the scaling of the rows and columns of L.

We see from (2.12) that if L and the partition are such that ρ is of order one, then the partitioned inverse method is normwise backward stable, that is, the computed solution $\widehat{x}$ solves a system obtained by making a tiny normwise perturbation to L. Using (2.12), the sparse backward error property noted earlier can be expressed as follows. Define $Z \in \mathbb{R}^{n\times n}$ by

$$z_{ij} = \begin{cases} 1, & \text{if } l_{ij} \ne 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.14}$$

Then, if each G_k is invertible in place,

$$(L+\Delta L)\widehat{x} = b, \qquad |\Delta L| \le \left(d_n u(m-1+\rho)\|L\|_\infty + O(u^2)\right) Z, \tag{2.15}$$

where the matrix inequality both bounds Δl_{ij} and shows that ΔL has the same sparsity structure as L.

Two useful upper bounds can be obtained for ρ. By examining the form of the matrix whose norm is the numerator in (2.13) it is easy to show that $\rho \le m \max_k \kappa_\infty(G_k)$. From the relation

$$\begin{aligned} G_k^{-1} &= G_{k+1}\dots G_m L^{-1} G_1 \dots G_{k-1} \\ &= G_{k+1}\dots G_m \begin{bmatrix} I_{i_k-1} & 0 \\ 0 & L^{-1}(i_k\colon n, i_k\colon n) \end{bmatrix}, \end{aligned}$$

we have $\|G_k^{-1}\|_\infty \le \max(1,\|L\|_\infty)\max(1,\|L^{-1}\|_\infty)$. As ρ is invariant under scalar multiplication of L, we can assume, without loss of generality, that $\|L\|_\infty = 1$, and hence we have, for all partitions,

$$\rho \le m\kappa_\infty(L).$$

We conclude that the normwise backward error for the partitioned inverse method is bounded by a multiple of $\kappa_\infty(L)u$. Although this bound may be very weak when L

is ill-conditioned, it shows that *if L is well-conditioned then the partitioned inverse method is guaranteed to be normwise backward stable.*

It is interesting to note that dependence of the backward error on the condition number occurs also in block LU factorization [5]. Another example of this dependence is a parallel triangular system solver analysed by Sameh and Brent [21], for which a backward error result with $\|\Delta L\| \leq c_n u \kappa^2(L)\|L\|$ is obtained. It seems to be a rule of thumb that if we attempt to improve the parallelism of Gaussian elimination or substitution we will achieve only conditional stability, with the backward error potentially proportional to some function of the condition number.

Now we turn to the forward error. One way to obtain a forward error bound is to expand the equation $\widehat{x} = (H_m + E_m)(H_{m-1} + E_{m-1}) \ldots (H_1 + E_1)b$, which follows from (2.7) and (2.10). For $m = 1$, this leads to the bound

$$\begin{aligned} \frac{\|x - \widehat{x}\|_\infty}{\|x\|_\infty} &\leq 2(n+1)u \frac{\| \, |L^{-1}||L||L^{-1}||b| \, \|_\infty}{\|x\|_\infty} + O(u^2) \\ &\leq 2(n+1)u \, \theta \| \, |L^{-1}||L| \, \|_\infty + O(u^2), \end{aligned} \tag{2.16}$$

where

$$\theta = \frac{\| \, |L^{-1}||b| \, \|_\infty}{\|L^{-1}b\|_\infty} \geq 1. \tag{2.17}$$

The scalar θ can be regarded as a measure of forward stability for the $L^{-1}b$ method. Note that θ is large only when there is a lot of cancellation through subtraction in the product $L^{-1}b$. Gill, Murray and Wright [10, Sec. 4.7.2] analyse the $L^{-1}b$ method under the simplifying assumption that the computed inverse is the correctly rounded inverse. Their forward error bound from a normwise analysis is proportional to $\kappa_\infty(L)(\|L^{-1}\|_\infty \|b\|_\infty / \|x\|_\infty)$, and so is consistent with our bound.

For general m, a more useful bound is obtained by manipulating the backward error result. Since $(L+\Delta L)\widehat{x} = b$ implies $|x - \widehat{x}| \leq |L^{-1}||\Delta L||\widehat{x}|$, we obtain from (2.11)

$$\begin{aligned} \frac{\|x - \widehat{x}\|_\infty}{\|x\|_\infty} \leq d_n u \frac{\left\| \left((m-1)|L^{-1}||L| + |L^{-1}|(\sum_{k=1}^m |G_k||H_k||G_k| - (m-1)I) \right)|x| \right\|_\infty}{\|x\|_\infty} \\ + O(u^2). \end{aligned} \tag{2.18}$$

The summation term precludes this bound from matching the ideal bound (2.2), but (2.18) does share with (2.2) the very desirable property that it is independent of the row scaling of the system. We can weaken (2.18) to obtain

$$\frac{\|x - \widehat{x}\|_\infty}{\|x\|_\infty} \leq d_n u (m - 1 + \rho) \kappa_\infty(L) + O(u^2), \tag{2.19}$$

where ρ is the growth factor in (2.13) (of course, this bound could have been obtained directly using (2.12)). Hence if ρ is of order one then the partitioned inverse method satisfies a bound of the form (2.3), that is, it is normwise forward stable.

3. Numerical experiments. We describe two numerical experiments that illustrate the error analysis and confirm the potential instability of the partitioned inverse method. Our experiments were performed in Matlab, which has unit roundoff $u \approx 1.1 \times 10^{-16}$. Statistics that we report include

$$\begin{aligned} \text{nberr} &= \min\{\epsilon : |b - L\widehat{x}| \leq \epsilon \|L\|_\infty e e^T |\widehat{x}|\}, \\ \text{sberr} &= \min\{\epsilon : |b - L\widehat{x}| \leq \epsilon \|L\|_\infty Z |\widehat{x}|\}, \\ \text{cberr} &= \min\{\epsilon : |b - L\widehat{x}| \leq \epsilon |L||\widehat{x}|\}, \end{aligned}$$

and

$$\text{ferr} = \frac{\|x - \widehat{x}\|_\infty}{\|x\|_\infty},$$

where $e = (1, 1, \ldots, 1)^T$ and Z is defined in (2.14). The quantity nberr is the usual normwise backward error, written in a way that shows its connection with the "sparse normwise backward error" sberr. From (2.12) it follows that, to first order, nberr $\leq$ $d_n u(m-1+\rho)$ and, if each G_k is invertible in place, sberr satisfies the same bound, by (2.15). The componentwise backward error cberr is $O(u)$ for substitution, by (2.1). We mention that in both experiments, modifying the backward errors nberr, sberr and cberr to include a b term (thus, allowing b to be perturbed in the definition of backward error) changes the backward errors by at most a factor 2.

In our first experiment $L = R^T$, where $V = QR$ is a QR factorization of the 15×15 Vandermonde matrix with (i, j) element $((j-1)/(n-1))^{i-1}$, and $b = Le$. This linear system is taken from [6, Sec. 4]. We solved the system $Lx = b$ using the partitioned inverse method with "fixed-width" partitions (1.3) having $i_{k+1} = i_k + p$, for several values of p. Results are reported in Table 3.1; since L is dense, nberr $\approx$ sberr, so we do not give the sberr values. We see that as p increases, the normwise backward error increases and the algorithm loses backward stability. In these examples, both (2.12) and (2.18) are a factor $\approx 10^3$ from being equalities for $p \geq 4$, and the bound $\rho \leq m \max_k \kappa_\infty(G_k)$ is clearly very weak. The ideal forward error bounds (2.2) and (2.3) are 6.43e-4 and 3.87e-3, respectively; ferr exceeds both values for $p \geq 6$, so the algorithm also loses forward stability. The quantity θ in (2.17) has the value $\theta =$ 3.60e11, so the bound (2.16) predicts forward instability when $m = 1$, but is a factor $\approx 10^8$ from being an equality. We also solved $Lx = c$, where $c = (-1, 1, -1, 1, \ldots)^T$, and the result for $p = 15$ is reported in the last line of Table 3.1. Here, $\theta = 1$, and, as predicted by (2.16), the algorithm performs in a forward stable manner; the tiny backward error is not predicted by our analysis.

In our second experiment L is a 20×20 matrix with 58 nonzeros; the entries and their locations were chosen randomly and then manipulated "by hand" to produce interesting behaviour. The inverse of L has 93 nonzeros. We solved the system $Lx = b$, where $b = Le$, using five different partitions. Partition A corresponds to forming $\widehat{x} = L^{-1} \times b$ ($m = 1$), partition B has $m = 2$ with $G_1 = L_1 L_2 \ldots L_{14}$ and $G_2 = L_{15} \ldots L_{20}$, partition C is a best no-fill partition, partition D is a best reordered partition, and partition E gives a variant of forward substitution ($m = n$). (Recall that partition D corresponds to a partition of a symmetric permutation of L that preserves its lower triangular structure.) In Table 3.2 we report the backward errors and ρ (the system is too ill-conditioned for us to determine the forward errors, but the computed solutions probably have no correct digits).

The results confirm two properties suggested by the analysis.
(1) For partitions in which the factors are not invertible in place (partitions A and B in the table), the sparse backward error can greatly exceed the normwise backward error.
(2) Even a best no-fill partition can yield sparse or componentwise backward errors appreciably larger than those for substitution.

We have been unable to construct a numerical example where the sparse backward error is large even when ρ is small, which the analysis suggests may be possible for partitions where the factors are not invertible in place. Our limited experience with the partitioned inverse method suggests that, like substitution, it frequently achieves

TABLE 3.1
Dense system $Lx = b$, $n = 15$.

$$\kappa_\infty(L) = 2.18\text{e}12,\ \mathrm{cond}(L,x) = 3.62\text{e}11$$

p	m	nberr	ferr	ρ	$\max_k \kappa_\infty(G_k)$
1	15	0.00	2.04e−6	3.00	8.38e6
2	8	4.98e−18	5.68e−6	2.65e1	1.55e7
4	4	5.77e−16	4.94e−4	1.49e3	9.51e8
6	3	2.14e−14	4.36e−3	3.62e4	9.51e8
8	2	9.10e−14	8.42e−3	5.68e5	1.03e11
10	2	4.73e−13	7.34e−3	2.04e6	1.41e10
12	2	6.63e−13	1.11e−2	2.72e6	3.83e9
15	1	6.60e−13	1.13e−2	2.78e6	2.18e12
	$Lx = c$, $\mathrm{cond}(L,x) = 3.90\text{e}4$				
15	1	9.07e−23	8.78e−11	2.78e6	2.18e12

TABLE 3.2
Sparse system $Lx = b$, $n = 20$.

$$\kappa_\infty(L) = 2.69\text{e}28,\ \mathrm{cond}(L,x) = 1.47\text{e}16$$

partition $(i_1, i_2, \ldots, i_{m+1})$	nberr	sberr	cberr	ρ
A: (1,21)	5.49e−9	4.26e−6	2.47e−1	7.36e19
B: (1,15,21)	1.83e−14	4.26e−6	9.48e−2	1.06e14
C: (1,2,3,4,11,12,13,19,21)	4.03e−24	1.55e−14	6.96e−14	5.67e3
D: (1,5,9,11,17,19,21)	4.93e−24	1.55e−14	6.96e−14	5.67e3
E: (1,2,3,...,21)	7.27e−24	3.80e−17	1.70e−16	3.00

surprisingly small forward and backward errors in practice. However, in view of the possible instability it is wise to compute one of the backward errors and make an *a posteriori* test for stability. Alternatively, if many right-hand sides are to be handled, it may be preferable to compute ρ or estimate its upper bound $m\kappa_\infty(L)$ before solving the systems. If any of these tests reveal or predict instability, substitution could be used instead.

Acknowledgements. We thank Des Higham for suggesting improvements to the manuscript.

REFERENCES

[1] F. L. Alvarado, A. Pothen, and R. S. Schreiber, *Highly parallel sparse triangular solution*, in Graph Theory and Sparse Matrix Computations, J. A. George, J. R. Gilbert, and J. W. H. Liu, eds., vol. 56 of IMA Volumes in Mathematics and its Applications, Springer-Verlag, New York, 1993, pp. 141–158.

[2] F. L. Alvarado and R. S. Schreiber, *Optimal parallel solution of sparse triangular systems*, SIAM J. Sci. Comput., 14 (1993), pp. 446–460.

[3] F. L. Alvarado, D. C. Yu, and R. Betancourt, *Partitioned sparse A^{-1} methods*, IEEE Trans. Power Systems, 5 (1990), pp. 452–458.

[4] J. W. Demmel, *Trading off parallelism and numerical stability*, in Linear Algebra for Large Scale and Real-Time Applications, M. S. Moonen, G. H. Golub, and B. L. D. Moor, eds., vol. 232 of NATO ASI Series E, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1993, pp. 49–68.

[5] J. W. Demmel, N. J. Higham, and R. S. Schreiber, *Stability of block LU factorization*, Numerical Linear Algebra with Applications, 2 (1995), pp. 173–190.

[6] J. J. Du Croz and N. J. Higham, *Stability of methods for matrix inversion*, IMA J. Numer. Anal., 12 (1992), pp. 1–19.

[7] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Oxford University Press, 1986.

[8] M. K. Enns, W. F. Tinney, and F. L. Alvarado, *Sparse matrix inverse factors*, IEEE Trans. Power Systems, 5 (1990), pp. 466–472.

[9] K. A. Gallivan, R. J. Plemmons, and A. H. Sameh, *Parallel algorithms for dense linear algebra computations*, SIAM Review, 32 (1990), pp. 54–135.

[10] P. E. Gill, W. Murray, and M. H. Wright, *Numerical Linear Algebra and Optimization*, vol. 1, Addison-Wesley, Reading, MA, USA, 1991.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, USA, second ed., 1989.

[12] M. T. Heath and C. H. Romine, *Parallel solution of triangular systems on distributed-memory multiprocessors*, SIAM J. Sci. Stat. Comput., 9 (1988), pp. 558–588.

[13] D. Heller, *A survey of parallel algorithms in numerical linear algebra*, SIAM Review, 20 (1978), pp. 740–777.

[14] N. J. Higham, *The accuracy of solutions to triangular systems*, SIAM J. Numer. Anal., 26 (1989), pp. 1252–1265.

[15] G. Li and T. F. Coleman, *A new method for solving triangular systems on distributed-memory message-passing multiprocessors*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 382–396.

[16] J. M. Ortega and R. G. Voigt, *Solution of partial differential equations on vector and parallel computers*, SIAM Review, 27 (1985), pp. 149–240.

[17] B. W. Peyton, A. Pothen, and X. Yuan, *Partitioning a chordal graph into transitive subgraphs for parallel sparse triangular solution*, Technical Report CS-92-55, Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada, Dec. 1992. To appear in Linear Algebra and Appl.

[18] ———, *A clique tree algorithm for partitioning a chordal graph into transitive subgraphs*, Technical Report CS-93-27, Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada, May 1993.

[19] A. Pothen and F. L. Alvarado, *A fast reordering algorithm for parallel sparse triangular solution*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 645–653.

[20] E. Rothberg and A. Gupta, *Parallel ICCG on a hierarchical memory multiprocessor—Addressing the triangular solve bottleneck*, Parallel Computing, 18 (1992), pp. 719–741.

[21] A. H. Sameh and R. P. Brent, *Solving triangular systems on a parallel computer*, SIAM J. Numer. Anal., 14 (1977), pp. 1101–1113.

[22] G. W. Stewart, *Introduction to Matrix Computations*, Academic Press, New York, 1973.