Preprint MCS–P229–0491, MCS Division, Argonne National Laboratory

STABLE PARALLEL ELIMINATION FOR BOUNDARY VALUE ODES*

STEPHEN J. WRIGHT†

Abstract. A parallelizable and vectorizable algorithm for solving linear algebraic systems arising from two-point boundary value ODEs is described. The method is equivalent to Gaussian elimination, with row partial pivoting, applied to a certain row- and column- reordered version of the usual almost-block-diagonal coefficient matrix. Analytical and numerical evidence is presented to show that the algorithm is stable. Results from implementation on a shared-memory multiprocessor and a vector processor are given. The approach can be extended to handle problems with multipoint and integral conditions, or algebraic parameters.

Key words. Parallel algorithms, two-point boundary value problems, Gaussian elimination, stability

AMS(MOS) subject classifications. 65L10, 65L20, 65W05

1. Introduction.

In a recent paper [11], we described a technique for solving the almost block-diagonal system of linear equations that arise in various algorithms for solving two-point boundary value problems, such as collocation, multiple shooting, and finite differencing. The advantage of the approach is that it can be adapted to make efficient use of parallel and vector computer architectures, while retaining the stability characteristics that are essential in robust software. The technique makes use of orthogonal transformations that take advantage of the special structure of the matrices.

In this report, we discuss a similar technique based on Gauss transformations. The new "structured elimination" technique and the structured orthogonal technique make identical use of the matrix structure, but the former requires substantially fewer floating-point operations. Timing results are given for some computational experiments on two advanced computer architectures. In Section 4, we investigate the stability of this structured elimination algorithm by relating it to a block factorization algorithm, which effectively decouples the increasing and decreasing fundamental modes when an exponential dichotomy is present and the problem is well conditioned. We show in Section 5 how the approach can be extended to handle problems with algebraic parameters or multipoint and integral side conditions.

Some of the details of the problem and the structured factorization approach have been discussed in [11] and hence are omitted here.

2. The Algorithm.

Consider the linear first-order two-point boundary value problem

$$y' = M(t)y + q(t), \qquad t \in [a,b], \qquad y \in \mathbf{R}^n, \tag{1}$$

$$B_a y(a) + B_b y(b) = d, \quad d \in \mathbf{R}^n. \tag{2}$$

Multiple shooting and finite differencing proceed by choosing a grid

$$a = t_1 < t_2 < \ldots < t_{k+1} = b$$

and seeking vectors s_i that approximate the true solution $y(t_i)$ for $i = 1, \ldots, k+1$. In both methods, linear systems of the following form must be solved for $s_1, s_2, \ldots, s_{k+1}$:

$$\begin{bmatrix} B_a & & & & B_b \\ A_1 & C_1 & & & \\ & A_2 & C_2 & & \\ & & \ddots & \ddots & \\ & & & A_k & C_k \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} d \\ f_1 \\ f_2 \\ \vdots \\ f_k \end{bmatrix}. \tag{3}$$

* This research was supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U. S. Department of Energy, under Contract W-31-109-Eng-38. A grant of computer time at the North Carolina Supercomputing Center is gratefully acknowledged.

† Argonne National Laboratory, 9700 South Cass Avenue, Argonne, Illinois 60439.

Here, $A_i, C_i \in R^{n\times n}$, $f_i \in R^n$. Collocation techniques give rise to similar systems, after condensation has been used to remove some of the parameters. In the case of separated end conditions, we can assume that B_a, B_b, and d have the form

$$B_a = \begin{bmatrix} \bar{B}_a \\ 0 \end{bmatrix}, \quad B_b = \begin{bmatrix} 0 \\ \bar{B}_b \end{bmatrix}, \quad d = \begin{bmatrix} d_a \\ d_b \end{bmatrix},$$

where $\bar{B}_a \in R^{l\times n}$ and $\bar{B}_b \in R^{(n-l)\times n}$. In this case, the system is usually reordered as

$$\begin{bmatrix} \bar{B}_a & & & & \\ A_1 & C_1 & & & \\ & A_2 & C_2 & & \\ & & \ddots & \ddots & \\ & & & A_k & C_k \\ & & & & \bar{B}_b \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_k \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} d_a \\ f_1 \\ f_2 \\ \vdots \\ f_k \\ d_b \end{bmatrix}. \tag{4}$$

Our algorithm starts, as in [11], by choosing an integer $P \leq k/2$, and a set of "separator" indices

$$0 = k_0 < k_1 < \ldots < k_P = k, \qquad k_{j+1} \geq k_j + 2, \;\; j = 0, \ldots, P.$$

Using these indices, we divide the coefficient matrix into P "slices," where slice j consists of rows $(k_{j-1}+1)n+1, \ldots, (k_j+1)n$, with the (structural) zero columns deleted. For example, the first slice is

$$\left[\begin{array}{ccccc|c} A_1 & C_1 & & & & f_1 \\ & A_2 & C_2 & & & f_2 \\ & & \ddots & \ddots & & \vdots \\ & & & A_{k_1} & C_{k_1} & f_{k_1} \end{array}\right]. \tag{5}$$

We now perform Gaussian elimination, with row partial pivoting, on the submatrix obtained by omitting the first and last block column for each slice. In (5), the first n steps of this process yield the factorization

$$\begin{bmatrix} U_1 \\ 0 \end{bmatrix} = \tilde{L}_{1,n}P_{1,n}\ldots\tilde{L}_{1,2}P_{1,2}\tilde{L}_{1,1}P_{1,1}\begin{bmatrix} C_1 \\ A_2 \end{bmatrix}, \tag{6}$$

where $P_{1,i} \in R^{2n\times 2n}$, $i = 1, \ldots, n$, are permutation matrices; $\tilde{L}_{1,i} \in R^{2n\times 2n}$, $i = 1, \ldots, n$, are Gauss transformations; and $U_1 \in R^{n\times n}$ is upper triangular. (The information needed to construct each $\tilde{L}_{1,i}$ can be stored in the locations formerly occupied by the eliminated elements of $[C_1^T \;\; A_2^T]^T$, while the permutation data for the whole process requires an integer array with dimension equal to that of the right-hand side of (3).) Applying the transformations across the first $2n$ rows of the matrix (5), we obtain an equivalent system whose augmented matrix is

$$\left[\begin{array}{cccccc|c} G_1 & U_1 & E_1 & & & & g_1 \\ \bar{G}_2 & 0 & \bar{C}_2 & & & & \bar{f}_2 \\ & & A_3 & C_3 & & & f_3 \\ & & & \ddots & \ddots & & \vdots \\ & & & & A_{k_1} & C_{k_1} & f_{k_1} \end{array}\right],$$

where

$$\left[\begin{array}{ccc|c} G_1 & U_1 & E_1 & g_1 \\ \bar{G}_2 & 0 & \bar{C}_2 & \bar{f}_2 \end{array}\right] = \tilde{L}_{1,n}P_{1,n}\ldots\tilde{L}_{1,2}P_{1,2}\tilde{L}_{1,1}P_{1,1}\left[\begin{array}{ccc|c} A_1 & C_1 & 0 & f_1 \\ 0 & A_2 & C_2 & f_2 \end{array}\right].$$

Repeating the process, for rows $n+1, \ldots, 3n$, we find permutation matrices $P_{2,i}$ and Gauss transformations $\tilde{L}_{2,i}$, $i = 1, \ldots, n$, such that

$$\left[\begin{array}{ccc|c} G_2 & R_2 & E_2 & g_2 \\ \bar{G}_3 & 0 & \bar{C}_3 & \bar{f}_3 \end{array}\right] = \tilde{L}_{2,n} P_{2,n} \ldots \tilde{L}_{2,2} P_{2,2} \tilde{L}_{2,1} P_{2,1} \left[\begin{array}{ccc|c} \bar{G}_2 & \bar{C}_2 & 0 & \bar{f}_2 \\ 0 & A_3 & C_3 & f_3 \end{array}\right].$$

Proceeding in this way for stages $3, \ldots, k_1 - 1$, we finally obtain the equivalent system

$$\left[\begin{array}{cccccc|c} G_1 & U_1 & E_1 & & & & g_1 \\ G_2 & & U_2 & E_2 & & & g_2 \\ \vdots & & & \ddots & \ddots & & \vdots \\ G_{k_1-1} & & & & U_{k_1-1} & E_{k_1-1} & g_{k_1-1} \\ \tilde{A}_1 & & & & & \tilde{C}_1 & \tilde{f}_1 \end{array}\right]. \tag{7}$$

A reduced system, involving only $(P+1)n$ variables and equations, can be formed by taking the last block row of each partition after the factorization above has been applied. As in [11], this system has the form

$$\begin{bmatrix} B_a & & & & B_b \\ \tilde{A}_1 & \tilde{C}_1 & & & \\ & \tilde{A}_2 & \tilde{C}_2 & & \\ & & \ddots & \ddots & \\ & & & \tilde{A}_P & \tilde{C}_P \end{bmatrix} \begin{bmatrix} s_1 \\ s_{k_1+1} \\ s_{k_2+1} \\ \vdots \\ s_{k_P+1} \end{bmatrix} = \begin{bmatrix} d \\ \tilde{f}_1 \\ \tilde{f}_2 \\ \vdots \\ \tilde{f}_P \end{bmatrix}. \tag{8}$$

Since the form of (8) is identical to that of the original system, the elimination procedure just described can be applied recursively. Eventually, a $2n \times 2n$ system of the form

$$\begin{bmatrix} B_a & B_b \\ \tilde{A} & \tilde{C} \end{bmatrix} \begin{bmatrix} s_1 \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} d \\ \tilde{f} \end{bmatrix} \tag{9}$$

is obtained. This can be solved by standard Gaussian elimination with partial pivoting. The "intermediate" unknowns can be recovered by back substitution, using the matrices U_i and E_i and the right-hand sides g_i which were obtained earlier in the factorization process.

Different variants of the overall algorithm can be distinguished by the number of levels used to obtain (9) from (3). A serial (one-level) algorithm would set $P = 1$ and hence go directly from (3) to (9). On a shared-memory machine, it may be appropriate to use a two-level algorithm, in which P is set equal to the number of available processors. The first stage of the reduction (which "compresses" (3) into (8)) can be carried out in parallel, while the compression of (8) to (9) can be performed on a single processor. On a vector processor, a "cyclic reduction" version, which uses $\log_2 k$ levels, is most appropriate. At the first level, we set $P = k/2$, so that each partition consists of just two block rows. A reduced system of approximately half the size of the original system is produced. The code can be written so that the innermost (vectorized) loops have length $k/2$. This procedure is applied recursively; at each level the size of the remaining system is halved. For more details, see [11].

For future reference, note that the elimination step (6) can be recast as follows: suppose that l of the pivot rows ($0 \le l \le n$) are drawn from the first n rows of the original matrix (i.e., the C_1 rows), so that $n - l$ of the pivots come from the A_2 rows. Then we can write

$$\begin{bmatrix} U_1 \\ 0 \end{bmatrix} = \begin{bmatrix} L_1^1 & 0 \\ L_1^2 & I \end{bmatrix} \begin{bmatrix} \bar{R}_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} P_a & P_b \\ P_b & P_a \end{bmatrix} \begin{bmatrix} \bar{O}_0 & 0 \\ 0 & \bar{O}_1 \end{bmatrix} \begin{bmatrix} C_1 \\ A_2 \end{bmatrix}, \tag{10}$$

where $\bar{R}_1$, $\bar{O}_0$ and $\bar{O}_1$ are $n \times n$ permutation matrices, L_1^1 is $n \times n$ and unit lower triangular, L_1^2 is $n \times n$, and

$$P_a = \begin{bmatrix} 0 & 0 \\ 0 & I_l \end{bmatrix}, \qquad P_b = \begin{bmatrix} I_{n-l} & 0 \\ 0 & 0 \end{bmatrix}. \tag{11}$$

TABLE 1
Operation counts and storage requirements for four algorithms, assuming separated end conditions (k = number of mesh points, n = dimension of y, p = number of left-hand end conditions, R = number of right-hand sides)

Algorithm	Operation Count	Storage
LU (row pivoting)	$k[\frac{5}{3}n^3 + 3pn^2 + R(4n^2 + 2pn)]$	$3kn^2$
`DECOMP/SOLVE`	$k[\frac{2}{3}n^3 + (4R + 5p)n^2 - 2np^2]$	$2kn^2$
Structured QR	$k[\frac{46}{3}n^3 + (15R + 30)n^2]$	$4kn^2$
Structured LU	$k[\frac{23}{3}n^3 + (8R - \frac{5}{2})n^2]$	$4kn^2$

TABLE 2
Operation counts and storage requirements for four algorithms, assuming nonseparated end conditions (k = number of mesh points, n = dimension of y, R = number of right-hand sides)

Algorithm	Operation Count	Storage
LU (row pivoting)	$k[\frac{23}{3}n^3 + 8Rn^2]$	$4kn^2$
`DECOMP/SOLVE`	$k[\frac{14}{3}n^3 + 4Rn^2]$	$3kn^2$
Structured QR	$k[\frac{46}{3}n^3 + (15R + 30)n^2]$	$4kn^2$
Structured LU	$k[\frac{23}{3}n^3 + (8R - \frac{5}{2})n^2]$	$4kn^2$

Note that

$$\begin{bmatrix} L_1^1 & 0 \\ L_1^2 & I \end{bmatrix} = \tilde{L}_{1,n}\tilde{L}_{1,n-1}\dots\tilde{L}_{1,1}.$$

The notation here is similar to that of Mattheij [9]. This is no accident, since the style of stability analysis in Section 4 is similar to that used in [9].

3. Computational Results. Variants of the structured elimination algorithm have been implemented on the Alliant FX/8 vector multiprocessor at Argonne National Laboratory and on the CRAY Y-MP at the North Carolina Supercomputing Center. In this section, we compare these codes with implementations of other algorithms for solution of the linear system arising from finite difference and shooting methods. These are

- a plain row partial pivoting code. The ordering (4) is used if the end conditions are separated. Otherwise, left-hand and coupled conditions are listed first, and right-hand conditions are listed at the end. We denote these codes by `ROWPP`.
- the `DECOMP` and `SOLVE` routines from the `PASVA` codes [7]. The `DECOMP` routine uses alternate row and column pivoting (as does the algorithm described in Varah [10]) but always eliminates by rows.
- the structured orthogonal factorization algorithm from [11]. These codes are generically denoted by `SQR`.

The structured elimination codes are referred to by the name `SLU`. The structured codes are subcategorized according to the number of levels. For example, the one-level, two-level, and cyclic reduction variants of the `SQR` code are named `SQR-1`, `SQR-2`, and `SQR-CR`, respectively.

Table 1 compares storage and operation counts for the four algorithms, assuming separated end conditions. In tabulating storage requirements, we assume that the multipliers and Householder vectors generated during the factorization are stored, for possible later use with different right-hand sides. For the two structured algorithms, the operation counts are independent of the number of recursive levels in the factorization process. Structured elimination requires about a third as many operations as structured orthogonal factorization, and they both have higher counts than the "serial" algorithms. However, operation counts are poor predictors of runtime, particularly on advanced computer architectures with their vectorization and pipelining capabilities. The structured algorithms are not as slow as Table 1 would suggest, even in serial mode. Table 2 contains the same information for the case in which the end conditions are completely coupled. The structured algorithms are indifferent to whether the end conditions are separated or nonseparated, whereas the operation counts for the serial algorithms increase markedly in the nonseparated case.

TABLE 3
Box method, ∞-norm error in the computed solution for Problem 1A

	$k = 32$	$k = 128$	$k = 1024$
`ROWPP, DECOMP, SQR-1, SLU-1`	.28(-1)	.17(-2)	.26(-4)

TABLE 4
Multiple shooting, ∞-norm error in the computed solution for Problem 1A

	$k = 8$	$k = 16$	$k = 64$
`ROWPP, DECOMP, SQR-1, SLU-1`	.17(-3)	.14(-4)	.35(-6)

The stability properties of the algorithms are tested by using them to solve two small problems based on the same ODE, but with different end conditions.

Problem 1A $a = 0$, $b = \pi$, $n = 3$,

$$y'(t) = \begin{bmatrix} 1 - 19\cos 2t & 0 & 1 + 19\sin 2t \\ 0 & 19 & 0 \\ -1 + 19\sin 2t & 0 & 1 + 19\cos 2t \end{bmatrix} y(t) + e^t \begin{bmatrix} -1 + 19(\cos 2t \; - \sin 2t) \\ -18 \\ 1 - 19(\cos 2t \; + \sin 2t) \end{bmatrix},$$

$$\begin{aligned} y_1(0) &= 1 \\ y_2(\pi) &= e^\pi \\ y_1(\pi) + 3y_3(\pi) &= 4e^\pi. \end{aligned}$$

The solution is $y(t) = e^t(1, 1, 1)^T$. A fundamental solution for this problem is

$$Y(t) = \begin{bmatrix} \sin t & 0 & -\cos t \\ 0 & 1 & 0 \\ \cos t & 0 & \sin t \end{bmatrix} \mathrm{diag}(e^{20t}, e^{19t}, e^{-18t}).$$

Problem 1B (Mattheij [9]) Same as Problem 1A, but with end conditions

$$\begin{aligned} y_1(0) &= 1 \\ y_3(0) + y_3(\pi) &= 1 + e^\pi \\ y_2(0) + y_2(\pi) &= 1 + e^\pi. \end{aligned}$$

Again, the solution is $y(t) = e^t(1, 1, 1)^T$.

The code **`dverk`** is used as the IVP solver in the multiple shooting code, with the global error tolerance set to 10^{-10}. The k subintervals are equally spaced in both codes. A midpoint-rule discretization is used in the finite-difference code (yielding the "box method"). Results from single-precision implementation on the CRAY Y-MP are reported.

Tables 3 and 4 show maximum errors in the computed solutions for Problem 1A. The behavior of the four codes is identical. Since the end conditions are separated, the two "serial" codes use the ordering (4); the matrix is now banded, and any reasonable partial pivoting strategy can be expected to be stable. The errors are consistent with estimates that take into account discretization error (in the case of the finite difference method) and conditioning of the problem.

Tables 5 and 6 show errors in the computed solutions for Problem 1B, which has coupled end conditions. In this respect, the behavior of the codes `ROWPP`, `SQR-1`, and `SLU-1` was indistinguishable. The code `DECOMP` does not properly decouple the modes, and so gives large errors.

TABLE 5
Box method, ∞-norm error in the computed solution for Problem 1B

	$k = 32$	$k = 128$	$k = 1024$
`ROWPP, SQR-1, SLU-1`	.28(-1)	.17(-2)	.26(-4)
`DECOMP`	.24(+2)	.24(+2)	.24(+2)

TABLE 6
Multiple shooting, ∞-norm error in the computed solution for Problem 1B

	$k = 8$	$k = 16$	$k = 64$
`ROWPP, SQR-1, SLU-1`	.17(-3)	.14(-4)	.35(-6)
`DECOMP`	.24(+2)	.24(+2)	.24(+2)

As expected, the compactification algorithm produces very large errors when applied to both 1A and 1B.

Four test problems are used to compare the relative execution speeds of the codes on the Alliant FX/8 and the CRAY Y-MP:

Problem 2 (Ascher and Chan [1]) $a = 0$, $b = 1$, $n = 2$,

$$y'(t) = \begin{bmatrix} -\lambda\cos 2\omega t & \omega + \lambda\sin 2\omega t \\ -\omega + \lambda\sin 2\omega t & \lambda\cos 2\omega t \end{bmatrix} y(t) + f(t), \qquad y_1(0) = 1, \quad y_1(1) = e,$$

with $f(t)$ chosen so that $y(t) = e^t(1,1)^T$.

Problem 3 (Brown and Lorenz [3]) $a = -1$, $b = 1$, $n = 4$,

$$\begin{aligned} -\epsilon y'' - \frac{t}{2}y' + \frac{t}{2}z' + z &= \epsilon\pi^2\cos\pi t + \frac{1}{2}\pi t\sin\pi t, \\ \epsilon z'' &= z \\ y(-1) = -1 \qquad & y(1) = e^{-2/\sqrt{\epsilon}} \\ z(-1) = 1 \qquad & z(1) = e^{-2/\sqrt{\epsilon}}. \end{aligned}$$

(We use $\epsilon = .001$.)

Problem 4A $a = 0$, $b = 1$, $n = 5$,

$$M(t) = \begin{bmatrix} -\lambda_1\cos 2\omega_1 t & 0 & \omega_1 + \lambda_1\sin 2\omega_1 t & 0 & 0 \\ 0 & -\lambda_2\cos 2\omega_2 t & 0 & \omega_2 + \lambda_2\sin 2\omega_2 t & 0 \\ -\omega_1 + \lambda_1\sin 2\omega_1 t & 0 & \lambda_1\cos 2\omega_1 t & 0 & 0 \\ 0 & -\omega_2 + \lambda_2\sin 2\omega_2 t & 0 & \lambda_2\cos 2\omega_2 t & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 \end{bmatrix},$$

$$q(t) = [q_1(t), q_2(t), q_3(t), q_4(t), -(\lambda_3 - 1)e^t]^T,$$

with boundary conditions

$$\begin{aligned} y_1(0) &= 1, \\ y_2(0) + 4y_5(0) &= 5, \end{aligned} \qquad \begin{aligned} y_1(1) &= e, \\ -y_3(1) + y_4(1) &= 0, \\ -4y_2(1) + 5y_5(1) &= e. \end{aligned}$$

TABLE 7
Alliant FX/8, one-processor timings for linear system solvers, $k = 1024$ (times in seconds)

Problem	ROWPP	DECOMP	SQR-1	SQR-CR	SLU-1	SLU-CR
2	.424	.667	.816	1.06	.576	.649
3	1.31	1.94	3.21	4.50	2.08	2.53
4A	1.94	2.71	5.42	7.79	3.48	4.40
4B	3.44	3.28	5.42	7.78	3.47	4.37

We use $\lambda_1 = 200$, $\lambda_2 = 50$, $\lambda_3 = 10$, $\omega_1 = 1$, and $\omega_2 = 25$. The components of $q(t)$ are chosen so that the solution is $e^t(1,1,1,1,1)^T$. This problem has a fundamental solution

$$Y(t) = \begin{bmatrix} \cos\omega_1 t & 0 & \sin\omega_1 t & 0 & 0 \\ 0 & \cos\omega_2 t & 0 & \sin\omega_2 t & 0 \\ -\sin\omega_1 t & 0 & \cos\omega_1 t & 0 & 0 \\ 0 & -\sin\omega_2 t & 0 & \cos\omega_2 t & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \operatorname{diag}\,(e^{-\lambda_1 t}, e^{-\lambda_2 t}, e^{\lambda_1 t}, e^{\lambda_2 t}, e^{\lambda_3 t})$$

and is obtained by replicating Problem 2.

Problem 4B Same ODE and solution as Problem 4A, except that three of the boundary conditions are coupled:

$$\begin{aligned} y_1(0) &= 1 \\ y_2(0) + 4y_5(0) + y_3(1) &= 5 + e \\ -5y_1(0) + y_1(1) &= -5 + e \\ 3y_2(0) - y_3(1) + y_4(1) &= 3 \\ -4y_2(1) + 5y_5(1) &= e. \end{aligned}$$

Tables 7 and 8 show execution times on one processor of the Alliant FX/8. The `-Og` option was used during compilation, so the vectorization capabilities of the processor were not used. Comparing these tables with the operation counts in Tables 1 and 2, we see that the time required by the `DECOMP` code is greater than expected, while the time required by `SQR-1` is less than expected. This result is at least partially due to the large amount of data manipulation performed by `DECOMP` (in the interests of economizing on storage) and the lack of such data manipulation in the `SQR` code, which avoids the need for pivoting altogether. The "cyclic reduction" codes take somewhat longer than their sequential counterparts, partly because their data access patterns are not localized. On problem 4B (the only one with some coupled end conditions), the time needed by `ROWPP` and `DECOMP` increases markedly over the time for Problem 4A. In fact, `SLU` becomes competitive here even without the benefit of vectorization or parallelization. In addition, we note that `DECOMP` always gives inaccurate answers for the coupled end-condition Problem 4B, because of its failure to decouple the increasing and decreasing fundamental modes.

Tables 9 and 10 give execution times for the codes when they have been compiled to run on eight processors of the Alliant without vectorization. The structured codes speed up as predicted, although the bottleneck of solving the reduced system on a single processor is particularly noticeable when $k = 128$. Defining speedup as the ratio of time for fastest serial algorithm to time for best parallel algorithm, we see from Table 11 that the `SLU-2` code demonstrates excellent efficiency for the larger problems, especially when the boundary conditions are not separated.

Results for vectorized implementations on the CRAY Y-MP are shown in Tables 12 and 13. The "cyclic reduction" versions of the structured codes vectorize very efficiently, as can be seen by comparing their execution times with the corresponding sequential versions. Speedups over the best serial codes fall off

Table 8
Alliant FX/8, one-processor timings for linear system solvers, $k = 128$ *(times in seconds)*

Problem	`ROWPP`	`DECOMP`	`SQR-1`	`SQR-CR`	`SLU-1`	`SLU-CR`
2	.065	.118	.119	.157	.085	.098
3	.172	.291	.416	.573	.273	.322
4A	.247	.399	.692	.989	.449	.543
4B	.445	.468	.697	.990	.446	.538

Table 9
Alliant FX/8, eight-processor timings for linear system solvers, $k = 1024$ *(times in seconds)*

Problem	`ROWPP`	`DECOMP`	`SQR-2`	`SLU-2`
2	.359	.695	.136	.104
3	.749	1.65	.466	.311
4A	.963	2.14	.765	.506
4B	1.37	2.79	.766	.505

Table 10
Alliant FX/8, eight-processor timings for linear system solvers, $k = 128$ *(times in seconds)*

Problem	`ROWPP`	`DECOMP`	`SQR-2`	`SLU-2`
2	.051	.114	.037	.035
3	.096	.242	.091	.069
4A	.124	.301	.143	.100
4B	.181	.387	.142	.100

Table 11
Alliant FX/8. Speedups, defined as time for fastest one-processor algorithm divided by time for fastest eight-processor algorithm

Problem	$k = 1024$	$k = 128$
2	4.08	1.86
3	4.21	2.49
4A	3.83	2.47
4B	6.50	4.45

TABLE 12
CRAY Y-MP, vectorized code on one processor, $k = 1024$ (times in milliseconds)

Problem	ROWPP	DECOMP	SQR-1	SQR-CR	SLU-1	SLU-CR
2	34.3	48.8	117.	10.8	83.0	7.3
3	84.7	134.	300.	51.6	194.	40.0
4A	111.	171.	415.	89.4	263.	71.4
4B	233.	245.	413.	89.5	261.	71.8

TABLE 13
CRAY Y-MP, vectorized code on one processor, $k = 128$ (times in milliseconds)

Problem	ROWPP	DECOMP	SQR-1	SQR-CR	SLU-1	SLU-CR
2	4.3	6.1	14.6	2.2	10.3	1.6
3	10.6	16.8	37.4	10.1	24.2	7.3
4A	13.8	21.4	51.8	17.3	32.8	12.5
4B	29.2	30.7	51.5	17.3	32.6	12.5

somewhat when the end conditions are separated and $n = 4$ or $n = 5$, since in these cases the codes `ROWPP` and `DECOMP` achieve a small amount of vectorization in the factorization of each of the blocks in the coefficient matrix in (4).

4. Stability of Structured Elimination.

The stability of the `SQR` algorithms has been examined in [11] by using standard tools for the analysis of orthogonal factorizations from the numerical linear algebra literature (see, for example, Golub and Van Loan [5]). Similar tools can be used to analyze the `SLU` algorithms, since *each variant of* `SLU` *is simply Gaussian elimination with row partial pivoting, applied to a row- and column-reordered version of the coefficient matrix in (3).* However, it is well known that, even with row pivoting, element growth can occur in the upper triangular factor which is exponential in the size of the matrix (in our case $(k + 1)n$). Although such growth is unlikely in practice, it is nevertheless desirable to use the structure of the matrix in (3) and the properties of the underlying problem (1), (2) to obtain a more realistic bound on the error growth in the method. We do this by using a style of analysis that was developed by Mattheij in a series of papers (see, for example [8], [9]). Our approach is first to describe a block factorization algorithm, which is similar to "stable compactification" (Section 4.2). This algorithm performs the required decoupling of the growing and decaying modes. Then, in Section 4.3, we show that the "serial" variant of `SLU` is essentially equivalent to this block algorithm under certain reasonable assumptions.

4.1. Dichotomy and Conditioning.

We start by briefly reviewing the salient points of the theory of dichotomy and conditioning, from Ascher, Mattheij, and Russell [2] and Mattheij [9].

The ODE (1) is said to have an *exponential dichotomy* if there exist real positive constants K, λ, and μ and an integer l with $0 \le l \le n$, such that, for some fundamental solution $Y(t)$ of (1),

$$\begin{aligned} \|Y(x)P_aY^{-1}(t)\| &\le Ke^{-\lambda(x-t)}, \qquad x \ge t, \\ \|Y(x)P_bY^{-1}(t)\| &\le Ke^{-\mu(t-x)}, \qquad t \ge x, \end{aligned} \tag{12}$$

where P_a and P_b are as in (11). Any other fundamental solution $Z(t)$ of (1) is related to $Y(t)$ by

$$Z(t) = Y(t)H = Y(t)\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix},$$

where $H \in R^{n\times n}$ is constant and nonsingular, $H_{11} \in R^{(n-l)\times(n-l)}$, etc.

Similarly, we can consider the linear recurrence given by the discrete system (3), namely,

$$A_is_i + C_is_{i+1} = f_i, \qquad i = 1, \ldots, k. \tag{13}$$

This recurrence is said to have an exponential dichotomy if there exist positive constants K, $\sigma > 1$, $\rho > 1$, and an integer l as above, such that for some fundamental solution $\{Y_i\}$ (with $A_iY_i + C_iY_{i+1} = 0$, $i = 1, \ldots, k$) we have

$$\begin{aligned} \|Y_i P_a Y_{j+1}^{-1}\| &\leq K\sigma^{j-i}, \qquad i \geq j+1, \\ \|Y_i P_b Y_j^{-1}\| &\leq K\rho^{i-j}, \qquad i \leq j. \end{aligned} \tag{14}$$

As in the continuous case, any other fundamental solution $\{Z_i\}$ of (13) can be related to $\{Y_i\}$ by

$$Z_i = Y_i H = Y_i \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}, \qquad i = 1, \ldots, k. \tag{15}$$

$\{Z_i\}$ is said to be *consistent* with $\{Y_i\}$ if H_{11} in (15) is nonsingular. Since the exact dichotomic solution $\{Y_i\}$ is not obtainable in practice, any consistent solution $\{Z_i\}$ can in theory be used to decouple the increasing and decreasing fundamental modes. It is sufficient to define *consistency constant* L by

$$L \stackrel{\text{def}}{=} \frac{\|Y_1^2 H_{21}\|}{\min_{\|u\|=1} \|Y_1^1 H_{11} u\|}$$

(where Y_1^1 and Y_1^2 are the first $(n-l)$ and last l columns of Y_1, respectively) and assume that L is of "moderate size."

It is known [2, p. 289] that if (3) is obtained by applying finite differencing to (1) with uniform mesh spacing $h \equiv t_{i+1} - t_i$, and if the ODE (1) has the dichotomy (12), then the recurrence (13) has the dichotomy (14) with $\sigma = e^{\lambda h}$ and $\rho = e^{\mu h}$.

From here until the end of the paper, we work solely with (3) and (13) and assume that the dichotomy (14) applies. We further assume that $C_i = -I$, $i = 1, \ldots, k$. This assumption is true for multiple shooting and the condensed form of the collocation equations; it can be obtained for finite differencing by multiplying (13) through by $-C_i^{-1}$. We can therefore rewrite (13) as

$$s_{i+1} = A_i s_i - f_i. \tag{16}$$

Suppose that we choose some orthogonal matrix $Q_0 \in R^{n \times n}$ and generate a sequence $\{Q_i\}$, $i = 0, \ldots, k$ of orthogonal matrices by using the formula

$$A_i Q_{i-1} = Q_i \Delta_i. \tag{17}$$

We require that Δ_i have the form

$$\Delta_i = \begin{bmatrix} \Delta_i^{11} & \Delta_i^{12} \\ 0 & \Delta_i^{22} \end{bmatrix}. \tag{18}$$

This could be obtained, for example, by defining Q_i, Δ_i as QR factors of $A_i Q_{i-1}$. If $\{Z_i\}$ is a fundamental solution of (16), we can use (17) to write

$$Z_{i+1} = A_i Z_i \Rightarrow (Q_i^T Z_{i+1}) = \Delta_i (Q_{i-1}^T Z_i),$$

so that $W_i = Q_{i-1}^T Z_i$, $W_1 = I$, defines a fundamental solution of the homogeneous recurrence

$$w_{i+1} = \Delta_i w_i, \qquad i = 1, \ldots, k.$$

It follows from (18) that each W_i has the form

$$W_i = \begin{bmatrix} W_i^{11} & W_i^{12} \\ 0 & W_i^{22} \end{bmatrix}, \qquad i = 1, \ldots, k+1. \tag{19}$$

Note, in particular, that

$$\begin{aligned} W_i^{11}(W_{j+1}^{11})^{-1} &= \prod_{r=i}^{j}(\Delta_r^{11})^{-1}, \qquad i \le j, \\ W_i^{22}(W_{j+1}^{22})^{-1} &= \Delta_{i-1}^{22}\Delta_{i-2}^{22}\dots\Delta_{j+1}^{22}, \qquad i \ge j+1. \end{aligned}$$

As a special case of [2, Theorem 6.30], we have the following:

THEOREM 4.1. *Suppose the dichotomy (14) holds, with fundamental solution $\{Y_i\}$, and that the fundamental solution $\{Z_i\}$ with initial condition $Z_1 = Q_0$ is consistent with $\{Y_i\}$, with L of moderate size. Then there are moderately sized constants $K_1 > 1$ and $K_2 > 1$ such that*

$$\begin{aligned} \|\prod_{r=i}^{j}(\Delta_r^{11})^{-1}\|_2 &\le K_1\rho^{i-j}, \qquad i \le j, \\ \|\Delta_{i-1}^{22}\Delta_{i-2}^{22}\dots\Delta_{j+1}^{22}\|_2 &\le K_2\sigma^{j-i}, \qquad i \ge j+1. \end{aligned}$$

4.2. A Structured Block Factorization. Given the assumption $C_i \equiv -I$, we can negate all but the boundary condition rows of (3) and perform an obvious row and column reordering to obtain the matrix

$$A = \left[\begin{array}{ccccc|cc} I & & & & & 0 & -A_1 \\ -A_2 & I & & & & 0 & 0 \\ & -A_3 & I & & & \vdots & \vdots \\ & & \ddots & \ddots & & & \\ & & & & I & 0 & 0 \\ \hline & & & & -A_k & I & 0 \\ & & & & & B_b & B_a \end{array}\right]. \tag{20}$$

In this section, we describe a block factorization algorithm for A and show that it gives a stable way of solving the system (3).

Using the orthogonal matrices $Q_0, Q_1, \dots, Q_k$ of the previous section, define

$$\begin{aligned} \tilde{R} &= \operatorname{diag}(Q_1^T, Q_2^T, \dots, Q_k^T, I) \\ \tilde{D} &= \operatorname{diag}(Q_1, Q_2, \dots, Q_k, Q_0). \end{aligned}$$

From (17) and (18) it follows that $\tilde{R}A\tilde{D}$ has the form

$$\tilde{R}A\tilde{D} = \left[\begin{array}{cccccccccc|cccc} I & 0 & & & & & & & & & & & -\Delta_1^{11} & -\Delta_1^{12} \\ 0 & I & & & & & & & & & & & 0 & -\Delta_1^{22} \\ -\Delta_2^{11} & -\Delta_2^{12} & I & 0 & & & & & & & & & & \\ 0 & -\Delta_2^{22} & 0 & I & & & & & & & & & & \\ & & -\Delta_3^{11} & -\Delta_3^{12} & I & 0 & & & & & & & & \\ & & 0 & -\Delta_3^{22} & 0 & I & & & & & & & & \\ & & & \ddots & \ddots & \ddots & \ddots & & & & & & & \\ & & & & & & & & I & 0 & & & & \\ & & & & & & & & 0 & I & & & & \\ \hline & & & & & & & & -\Delta_k^{11} & -\Delta_k^{12} & I & 0 & 0 & 0 \\ & & & & & & & & 0 & -\Delta_k^{22} & 0 & I & 0 & 0 \\ & & & & & & & & & & \multicolumn{2}{c}{(B_bQ_k)} & \multicolumn{2}{c}{(B_aQ_0)} \end{array}\right].$$

Using the $n \times n$ permutation matrices P_a and P_b from (11) as building blocks, we define $(k+1)n$-dimensional permutation matrices P_i, $i = 1, \ldots, (k-1)$, by

$$P_i = \left[\begin{array}{ccccccc}
I & & & & & & \\
 & \ddots & & & & & \\
 & & I & & & & \\
 & & & P_a & P_b \text{ (block row } i) & & \\
 & & & P_b & P_a \text{ (block row } i+1) & & \\
 & & & & I & & \\
 & & & & & \ddots & \\
 & & & & & & I
\end{array}\right].$$

Let

$$P = P_{k-1} P_{k-2} \ldots P_1.$$

Applying these permutations to the rows of $\tilde{R}A\tilde{D}$, we obtain

$$P\tilde{R}A\tilde{D} =$$
$$\left[\begin{array}{cccccccc|cccc}
-\Delta_2^{11} & -\Delta_2^{12} & I & 0 & & & & & & & 0 & 0 \\
0 & I & 0 & 0 & & & & & & & 0 & -\Delta_1^{22} \\
 & 0 & -\Delta_3^{11} & -\Delta_3^{12} & I & & & & & & & \\
 & -\Delta_2^{22} & 0 & I & 0 & & & & & & & \\
 & & & 0 & -\Delta_4^{11} & -\Delta_4^{12} & I & & & & & \\
 & & & -\Delta_3^{22} & 0 & I & 0 & & & & & \\
 & & & & \ddots & \ddots & \ddots & \ddots & & & & \\
 & & & & & & -\Delta_k^{11} & -\Delta_k^{12} & I & 0 & & \\
 & & & & & & 0 & I & 0 & 0 & & \\
\hline
I & 0 & & & & & 0 & 0 & 0 & 0 & -\Delta_1^{11} & -\Delta_1^{12} \\
0 & 0 & & & & & 0 & -\Delta_k^{22} & 0 & I & 0 & 0 \\
 & & & & & & & & \multicolumn{2}{c}{(B_b Q_k)} & \multicolumn{2}{c}{(B_a Q_0)}
\end{array}\right].$$

Now, without further pivoting, we can perform block elimination in an obvious way, stopping when the final $2n \times 2n$ submatrix is reached. We thus obtain

$$P\tilde{R}A\tilde{D} = \tilde{L}'\tilde{U}',$$

where

$$\tilde{L}' = \left[\begin{array}{ccccc|cc}
I & & & & & & \\
\tilde{L}_1 & I & & & & & \\
 & \tilde{L}_2 & I & & & & \\
 & & \ddots & \ddots & & & \\
 & & & \tilde{L}_{k-2} & I & & \\
\hline
\tilde{H}_1 & & & \tilde{H}_{k-2} & \hat{H}_{k-1} & I & \\
 & & & & & & I
\end{array}\right],$$

$$\tilde{U}' = \left[\begin{array}{cccc|cc}
\tilde{U}_1 & \tilde{E}_1 & & & & \tilde{G}_1 \\
 & \tilde{U}_2 & \tilde{E}_2 & & & \tilde{G}_2 \\
 & & \ddots & \ddots & & \vdots \\
 & & & \tilde{U}_{k-1} & \tilde{E}_{k-1} & \tilde{G}_{k-1} \\
\hline
 & & & & \hat{R}_1 & \hat{R}_2 \\
 & & & & B_b Q_k & B_a Q_0
\end{array}\right],$$

$$\tilde{L}_i = \begin{bmatrix} 0 & 0 \\ 0 & -\Delta_{i+1}^{22} \end{bmatrix}, \qquad \tilde{U}_i = \begin{bmatrix} -\Delta_{i+1}^{11} & -\Delta_{i+1}^{12} \\ 0 & I \end{bmatrix}, \qquad \tilde{E}_i = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix},$$

$$\tilde{H}_i = \begin{bmatrix} -\Delta_1^{11} S_{i+1}^{-1} & -\Delta_1^{11} S_{i+1}^{-1} \Delta_{i+1}^{12} \\ 0 & 0 \end{bmatrix}, \ \tilde{G}_i = \begin{bmatrix} 0 & 0 \\ 0 & -T_i \end{bmatrix} \quad i = 1, \ldots, k-1,$$

$$\hat{H}_{k-1} = \tilde{H}_{k-1} + \tilde{L}_{k-1},$$

$$\hat{R}_1 = \begin{bmatrix} -\Delta_1^{11} S_k^{-1} & 0 \\ 0 & I \end{bmatrix}, \qquad \hat{R}_2 = \begin{bmatrix} -\Delta_1^{11} & -\Delta_1^{11} X \\ 0 & -T_k \end{bmatrix},$$

$$S_i = \Delta_i^{11} \Delta_{i-1}^{11} \ldots \Delta_1^{11}, \qquad T_i = \Delta_i^{22} \Delta_{i-1}^{22} \ldots \Delta_1^{22},$$

$$X = S_1^{-1} \Delta_1^{12} - \sum_{j=1}^{k-1} S_{j+1}^{-1} \Delta_{j+1}^{12} T_j.$$

Finally, we can find a $2n \times 2n$ permutation matrix $\tilde{\Pi}$, unit lower triangular $\tilde{\tilde{L}}$ and upper triangular $\tilde{\tilde{U}}$ such that

$$\tilde{\tilde{L}} \tilde{\tilde{U}} = \tilde{\Pi} \begin{bmatrix} \hat{R}_1 & \hat{R}_2 \\ B_b Q_k & B_a Q_0 \end{bmatrix}.$$

Incorporating this into the overall factorization, we obtain

$$\begin{bmatrix} I & 0 \\ 0 & \tilde{\Pi} \end{bmatrix} P \tilde{R} A \tilde{D} = \tilde{L} \tilde{U}, \tag{21}$$

where

$$\tilde{L} = \left[\begin{array}{ccccc|c}
I & & & & & \\
\tilde{L}_1 & I & & & & \\
 & \tilde{L}_2 & I & & & \\
 & & \ddots & \ddots & & \\
 & & & \tilde{L}_{k-2} & I & \\
\hline
\tilde{H}_1^1 & \tilde{H}_2^1 & \ldots & \tilde{H}_{k-2}^1 & \hat{H}_{k-1}^1 & \\
\tilde{H}_1^2 & \tilde{H}_2^2 & \ldots & \tilde{H}_{k-2}^2 & \hat{H}_{k-1}^2 & \tilde{\tilde{L}}
\end{array}\right]$$

$$\tilde{U} = \left[\begin{array}{cccc|cc} \tilde{U}_1 & \tilde{E}_1 & & & & \tilde{G}_1 \\ & \tilde{U}_2 & \tilde{E}_2 & & & \tilde{G}_2 \\ & & \ddots & \ddots & & \vdots \\ & & & \tilde{U}_{k-1} & \tilde{E}_{k-1} & \tilde{G}_{k-1} \\ \hline & & & & \multicolumn{2}{c}{\tilde{\tilde{U}}} \end{array}\right],$$

and

$$\left[\begin{array}{c} \tilde{H}_i^1 \\ \tilde{H}_i^2 \end{array}\right] = \tilde{\Pi}\left[\begin{array}{c} \tilde{H}_i \\ 0 \end{array}\right], i = 1, \ldots, k-1, \qquad \left[\begin{array}{c} \hat{H}_{k-1}^1 \\ \hat{H}_{k-1}^2 \end{array}\right] = \tilde{\Pi}\left[\begin{array}{c} \hat{H}_{k-1} \\ 0 \end{array}\right].$$

The entire process is almost identical to stabilized compactification (as described in [2, pp. 157–161]), the only difference being in the handling of the final $2n \times 2n$ block. Stability of the factorization (21) follows immediately from this relationship. Alternatively, from the standard viewpoint of numerical linear algebra, stability follows from the fact that element growth in the $\tilde{L}$ and $\tilde{U}$ factors is bounded by reasonable quantities. To see this, define

$$\tau \stackrel{\text{def}}{=} \max(1, \|A_1\|_2, \ldots, \|A_k\|_2, \|B_a\|_2, \|B_b\|_2),$$

and note that

$$\tau \geq \|A_i\|_2 = \|\Delta_i\|_2 \geq \|\Delta_i^{12}\|_2,$$
$$\tau \geq \|\Delta_i^{11}\|_2, \quad \tau \geq \|\Delta_i^{22}\|_2, \quad i = 1, \ldots, k.$$

We have from Theorem 4.1 that

$$\begin{aligned} \|S_i^{-1}\|_2 &\leq (\rho K_1)\rho^{-i}, \\ \|T_i\|_2 &\leq (K_2/\sigma)\sigma^{-i}. \end{aligned}$$

Hence

$$\begin{aligned} \sum_{i=1}^{k-2} \|\tilde{H}_i\|_2 + \|\hat{H}_{k-1}\|_2 &\leq \sum_{i=1}^{k-1} \left[\|\Delta_1^{11} S_{i+1}^{-1}\|_2 + \|\Delta_1^{11} S_{i+1}^{-1} \Delta_{i+1}^{12}\|_2\right] + \|\Delta_k^{22}\|_2 \\ &\leq (\tau + \tau^2) \sum_{i=1}^{k-1} \|S_{i+1}^{-1}\|_2 + \tau \\ &\leq (\tau + \tau^2) K_1 \frac{\rho}{\rho - 1} + \tau. \end{aligned}$$

In addition,

$$\begin{aligned} \|X\|_2 &\leq (K_1\rho)(K_2/\sigma)\tau\left[\rho^{-1} + \rho^{-2}\sigma^{-1} + \ldots + \rho^{-k}\sigma^{-k+1}\right] \leq \frac{\rho K_1 K_2 \tau}{\rho\sigma - 1}, \\ \|\tilde{L}_i\|_2 &\leq \tau, \\ \|\tilde{U}_i\|_2 &\leq 3\tau, \\ \|\tilde{E}_i\|_2 &= 1, \\ \|\tilde{G}_i\|_2 &\leq (K_2/\sigma)\sigma^{-i}. \end{aligned}$$

Note that if $\sigma = e^{\lambda h}$ and $\rho = e^{\mu h}$, where h is a finite difference mesh width, and if $\lambda h \ll 1$ and $\mu h \ll 1$, then terms such as $1/(\rho - 1)$ and $1/(\rho\sigma - 1)$ are $O(h^{-1})$.

For future reference, we state the result as a theorem:

THEOREM 4.2. *Provided that the assumptions of Theorem 4.1 are satisfied, the factorization (21) is stable.*

4.3. Relating the Block Factorization to `SLU`. We now examine the "serial" version of structured elimination and show how it is related to a block factorization for a particular choice of Q_0. This relationship can be used to prove the stability of `SLU`.

Recall that (10) and (11) depict the first of $k+1$ stages of the factorization process. Note that in this stage, a total of $n-l$ rows were exchanged between the first and second block rows. Throughout the remainder of the paper, we make the following assumption:

Assumption 1. *The ODE (1) has an exponential dichotomy, with l decreasing fundamental modes ($1 \le l \le n$). At each stage i of the structured elimination algorithm (for $i = 1, \ldots, k$), exactly $n-l$ rows are exchanged, by row pivoting, between the i-th block row and the $(i+1)$-st block row. Moreover, the collection of $n-l$ rows that is swapped out of the first block row is passed down, intact, through block rows $i = 2, 3, \ldots, k$.*

This assumption was always observed to hold for the examples reported in Section 3. It means that if, for example, row 1 of block 1 is swapped into block 2 during the first stage, then this same row is eventually pivoted to the bottom of the matrix, that is, to somewhere in the k-th or $(k+1)$-st block rows. We stress that this does *not* mean that "reimbedding," as described in Dieci, Osborne, and Russell [4] and Keller and Lentini [6], does not occur. On the contrary, in all the test examples, we observe reimbedding in the form of occasional changes in the internal pivot sequence from one stage to the next. The last statement of Assumption 1 refers to the collection of rows that are exchanged between successive stages, rather than the pivot sequence within a stage.

Using Assumption 1, we can view stage i of the factorization process (for $i = 1, 2, \ldots, k-1$) as a premultiplication of the remaining submatrix by

$$L_i R_i P_i O_i,$$

where, for $i = 1, 2, \ldots, k-1$, P_i is as defined in Section 4.2, and

$$L_i = \begin{bmatrix} I &&&&&&& \\ & \ddots &&&&&& \\ && I &&&&& \\ &&& L_i^1 && \text{(block row } i) && \\ &&& L_i^2 & I & \text{(block row } i+1) && \\ &&&&& \ddots && \\ &&&&&& \ddots & \\ &&&&&&& I \end{bmatrix},$$

$$R_i = \begin{bmatrix} I &&&&&& \\ & \ddots &&&&& \\ && I &&&& \\ &&& \bar{R}_i & \text{(block row } i) && \\ &&&& I && \\ &&&&& \ddots & \\ &&&&&& I \end{bmatrix},$$

$$O_i = \begin{bmatrix} I & & & & & & \\ & \ddots & & & & & \\ & & I & & & & \\ & & & \bar{O}_i & \text{(block row } i+1) & & \\ & & & & I & & \\ & & & & & \ddots & \\ & & & & & & I \end{bmatrix}.$$

Here, L_i^1 is unit lower triangular, and $\bar{R}_i$ and $\bar{O}_i$ are permutation matrices. Before the very first stage, the matrix A is multiplied by some initial permutation $O_0 = \operatorname{diag}(\bar{O}_0, I, I, \ldots, I)$. For the final stage, in which the remaining $2n \times 2n$ block is factored, we define a $2n \times 2n$ permutation matrix $\bar{R}_k$, and premultiply by $L_k R_k$, where

$$L_k = \left[\begin{array}{ccc|c} I & & & \\ & \ddots & & \\ & & I & \\ \hline & & & L_k^1 \end{array}\right],$$

$$R_k = \left[\begin{array}{ccc|c} I & & & \\ & \ddots & & \\ & & I & \\ \hline & & & \bar{R}_k \end{array}\right],$$

and L_k^1 is $2n \times 2n$ unit lower triangular. At the end of the process, a block upper triangular matrix U is obtained, with

$$(L_k R_k)(L_{k-1} R_{k-1} P_{k-1} O_{k-1}) \ldots (L_1 R_1 P_1 O_1) O_0 A = U, \tag{22}$$

where A is as in (20).

We now rework (22) so that it is more directly comparable with the block factorization (21). To start with, simple manipulation involving the permutation blocks in R_i and O_i yields

$$\begin{array}{rc} & (L_k R_k)(L_{k-1} R_{k-1} P_{k-1} O_{k-1}) \ldots (L_1 R_1 P_1 O_1) O_0 \\ = & R(\bar{L}_k \bar{\bar{L}}_{k-1} P_{k-1} \bar{\bar{L}}_{k-2} P_{k-2} \ldots P_2 \bar{\bar{L}}_1 P_1) O, \end{array} \tag{23}$$

where

$$\begin{array}{rcl} R & = & \operatorname{diag}(\bar{R}_1, \bar{R}_2, \ldots, \bar{R}_{k-1}, \bar{R}_k) \\ O & = & \operatorname{diag}(\bar{O}_0, \bar{O}_1, \bar{O}_2, \ldots, \bar{O}_{k-1}, I), \end{array}$$

and, for $i = 1, \ldots, k-1$, $\bar{\bar{L}}_i$ has the form

$$\bar{\bar{L}}_i = \begin{bmatrix} I & & & & & & \\ & \ddots & & & & & \\ & & I & & & & \\ & & & \bar{L}_i^1 & & & \\ & & & \bar{L}_i^2 & I & & \\ & & & & & I & \\ & & & & & & \ddots & \\ & & & & & & & I \end{bmatrix},$$

where

$$\begin{aligned}\bar{L}_i^1 &= \bar{R}_i^T L_i^1 \bar{R}_i \\ \bar{L}_i^2 &= L_i^2 \bar{R}_i.\end{aligned}$$

For the final elimination matrix, we have used the notation

$$\bar{L}_k = R_k^T L_k R_k = \mathrm{diag}\,(I, I, \ldots, I, \bar{L}_k^1), \qquad \text{where} \qquad \bar{L}_k^1 = \bar{R}_k^T L_k^1 \bar{R}_k.$$

We can then use the following result, which is similar to property 8.56 of Mattheij [9], to isolate the "row exchange" matrices $P_1, \ldots, P_k$. Its proof is simple and hence is not included here.

LEMMA 4.3. *For $i = 1, \ldots, k-2$,*

$$P_{k-1}P_{k-2}\ldots P_{i+1}\bar{\bar{L}}_i = \bar{L}_i P_{k-1}P_{k-2}\ldots P_{i+1},$$

where

$$\bar{L}_i = \left[\begin{array}{cccccccccc} I & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & I & & & & & & & \\ & & & \bar{L}_i^1 & & & & & & \\ & & & P_a\bar{L}_i^2 & I & & & & & \\ & & & & & I & & & & \\ & & & & & & \ddots & & & \\ & & & & & & & I & & \\ & & & P_b\bar{L}_i^2 & 0 & 0 & & \ldots & I & 0 \\ & & & & & & & & & I \end{array}\right].$$

Lemma 4.3 can be used, together with (22) and (23) and the definition of P, to yield

$$RL^{-1}POA = U, \tag{24}$$

where, defining $\bar{L}_k = \bar{\bar{L}}_{k-1}$ for consistency, we have

$$L^{-1} = \bar{L}_k \bar{L}_{k-1} \bar{L}_{k-2} \ldots \bar{L}_1.$$

Apart from some permutations internal to the blocks, the back-substitution part of the solution process can be viewed as premultiplication by $L^{-1}P$. Suppose the vector $z \in R^{(k+1)n}$ is partitioned as

$$z = (z_b^1, z_a^1, z_b^2, z_a^2, \ldots, z_b^k, z_a^k, z_b^{k+1}, z_a^{k+1})^T,$$

where $z_b^i \in R^{n-l}$ and $z_a^i \in R^l$. Then

$$Pz = (z_b^2, z_a^1, z_b^3, z_a^2, \ldots, z_b^k, z_a^{k-1}, z_b^1, z_a^k, z_b^{k+1}, z_a^{k+1})^T.$$

For $i = 2, \ldots, k-1$, the subvector (z_b^{i+1}, z_a^i) is affected only by premultiplication by $\bar{L}_{i-1}$ and $\bar{L}_i$. Meanwhile, the next-to-last subvector (z_b^1, z_a^k) is affected by *every* $\bar{L}_i$. We use the following notation:

- (z_b^{i+1}, z_a^i) is transformed to $(\bar{z}_b^{i+1}, \bar{z}_a^i)$ when premultiplication by $\bar{L}_{i-1}$ is performed;
- $({}^i z_b^1, {}^i z_a^k)$ denotes the transformed version of (z_b^1, z_a^k), after $\bar{L}_1, \ldots, \bar{L}_i$ have been applied.

For later reference, we also set

$$\bar{z}^i = \bar{L}_i^1 \begin{bmatrix} \bar{z}_b^{i+1} \\ \bar{z}_a^i \end{bmatrix}, \quad i = 1, \ldots, k-1, \qquad \begin{bmatrix} \bar{z}^k \\ \bar{z}^{k+1} \end{bmatrix} = \bar{L}_k^1 \begin{bmatrix} {}^{k-1}z_b^1 \\ {}^{k-1}z_a^k \\ z_b^{k+1} \\ z_a^{k+1} \end{bmatrix}, \tag{25}$$

so that $\tilde{z} = (\tilde{z}^1, \ldots, \tilde{z}^{k+1})^T$ is the final outcome of the application of $L^{-1}P$. Note that since the $\bar{L}_i^1$ $(i = 1, \ldots, k)$ are row and column permutations of unit lower triangular matrices, they are certainly nonsingular.

Using the notation of Lemma 4.3, we have, upon premultiplication by $\bar{L}_{i-1}$, that

$$\begin{aligned} \left[\begin{array}{c} \bar{z}_b^{i+1} \\ \bar{z}_a^i \end{array}\right] &= P_a \bar{L}_{i-1}^2 \left[\begin{array}{c} \bar{z}_b^i \\ \bar{z}_a^{i-1} \end{array}\right] + \left[\begin{array}{c} z_b^{i+1} \\ z_a^i \end{array}\right] \\ &= \left[\begin{array}{cc} 0 & 0 \\ (\bar{L}_{i-1}^2)_{21} & (\bar{L}_{i-1}^2)_{22} \end{array}\right] \left[\begin{array}{c} \bar{z}_b^i \\ \bar{z}_a^{i-1} \end{array}\right] + \left[\begin{array}{c} z_b^{i+1} \\ z_a^i \end{array}\right]. \end{aligned} \tag{26}$$

(Here, as earlier, we have used $(H)_{11}$ to denote the leading $(n-l) \times (n-l)$ submatrix of a generic $n \times n$ matrix H, with the remaining blocks H_{12}, H_{21}, and H_{22} defined accordingly.) Examining the last block row, we see that

$$\begin{aligned} \left[\begin{array}{c} {}^{(i-1)}z_b^1 \\ {}^{(i-1)}z_a^{k+1} \end{array}\right] &= P_b \bar{L}_{i-1}^2 \left[\begin{array}{c} \bar{z}_b^i \\ \bar{z}_a^{i-1} \end{array}\right] + \left[\begin{array}{c} {}^{(i-2)}z_b^1 \\ {}^{(i-2)}z_a^k \end{array}\right] \\ &= \left[\begin{array}{cc} (\bar{L}_{i-1}^2)_{11} & (\bar{L}_{i-1}^2)_{12} \\ 0 & 0 \end{array}\right] \left[\begin{array}{c} \bar{z}_b^i \\ \bar{z}_a^{i-1} \end{array}\right] + \left[\begin{array}{c} {}^{(i-2)}z_b^1 \\ {}^{(i-2)}z_a^k \end{array}\right]. \end{aligned} \tag{27}$$

By combining and assembling the nontrivial parts of (26) and (27), we find that

$$\left[\begin{array}{c} {}^{(i-1)}z_b^1 \\ \bar{z}_a^i \end{array}\right] = \left[\begin{array}{cc} I & (\bar{L}_{i-1}^2)_{12} \\ 0 & (\bar{L}_{i-1}^2)_{22} \end{array}\right] \left[\begin{array}{c} {}^{(i-2)}z_b^1 \\ \bar{z}_a^{i-1} \end{array}\right] + \left[\begin{array}{cc} (\bar{L}_{i-1}^2)_{11} & 0 \\ (\bar{L}_{i-1}^2)_{21} & I \end{array}\right] \left[\begin{array}{c} z_b^i \\ z_a^i \end{array}\right]. \tag{28}$$

We now show that (28) is a stable recurrence by relating it to the block factorization described in Section 4.2. The block factorization can be initialized in the following way: Define $Q_1 = \bar{O}_0^T$, and then choose Q_0 in such a way that $Q_1^T A_1 Q_0$ is upper triangular. The remaining matrices $Q_2, \ldots, Q_k$ can then be obtained as in (17) and (18), leading to the factorization (21). We need the following assumption:

ASSUMPTION 2. *The matrix Q_0 obtained by the procedure just described induces a consistent fundamental solution $\{Z_i\}$ (with $Z_1 = Q_0$) of the recurrence (16), with a consistency constant of moderate size.*

Intuitively, it seems almost certain that this assumption will be satisfied. The first $n-l$ rows of the reordered matrix $Q_1^T A_1 = \bar{O}_0 A_1$ tend to correspond to the "larger" rows that are exchanged with the second stage. Postmultiplication by Q_0 yields Δ_1^{11} and Δ_1^{22}, which can be expected to satisfy the inequalities of Theorem 4.1, with K_1 and K_2 not much larger than 1.

Comparing (21) with (24), we find that

$$RL^{-1} \left(PO\tilde{R}^T P^T \left[\begin{array}{cc} I & 0 \\ 0 & \tilde{\Pi}^T \end{array}\right] \right) \tilde{L} = U\tilde{D}\tilde{U}^{-1}. \tag{29}$$

Clearly, by inspection of the structures of $\tilde{D}$, U and $\tilde{U}$, we see that both sides of this equation are block upper triangular matrices. We examine the structure of the left-hand side more closely. Using the notation

$$V_i = \bar{O}_{i-1} Q_i, \quad i = 1, 3, \ldots, k, \qquad (V_1 = \bar{O}_0 Q_1 = I), \qquad V_{k+1} = I,$$

we have $O\tilde{R}^T = \mathrm{diag}\,(V_1, V_2, \ldots, V_k, V_{k+1})$. Hence

$$PO\tilde{R}^T P^T = \tag{30}$$

$$\left[\begin{array}{cccccccccccc}
(V_2)_{11} & 0 & 0 & (V_2)_{12} & & & & & & & & \\
0 & (V_1)_{22} & 0 & 0 & & & & & & & & \\
0 & 0 & (V_3)_{11} & 0 & 0 & (V_3)_{12} & & & & & & \\
(V_2)_{21} & 0 & 0 & (V_2)_{22} & 0 & 0 & & & & & & \\
 & & 0 & 0 & & & & & & & & \\
 & & (V_3)_{21} & 0 & & \ddots & \ddots & & & & & \\
 & & & & \ddots & \ddots & \ddots & & & & & \\
 & & & & & & & (V_k)_{11} & 0 & 0 & (V_k)_{12} & \\
 & & & & & & & 0 & (V_{k-1})_{22} & 0 & 0 & \\
 & & & & & & & 0 & 0 & (V_1)_{11} & 0 & \\
 & & & & & & & (V_k)_{21} & 0 & 0 & (V_k)_{22} & \\
 & & & & & & & & & & & I
\end{array}\right],$$

where the lower right identity block is $n \times n$. Further, from the definition of $\tilde{L}$ in Section 4.2, it is easy to see that

$$\left[\begin{array}{cc} I & 0 \\ 0 & \tilde{\Pi}^T \end{array}\right] \tilde{L} = \left[\begin{array}{ccccc|c}
I & & & & & \\
\tilde{L}_1 & I & & & & \\
 & \tilde{L}_2 & I & & & \\
 & & \ddots & \ddots & & \\
 & & & & I & \\
\hline
\tilde{H}_1 & \tilde{H}_2 & \dots & \tilde{H}_{k-2} & \hat{H}_{k-1} & \tilde{\Pi}^T \tilde{\tilde{L}} \\
0 & 0 & \dots & 0 & 0 &
\end{array}\right]. \tag{31}$$

When the matrices (30) and (31) are multiplied together, we obtain the following block lower Hessenberg matrix, which we denote by J^0:

$$J^0 = (PO\tilde{R}^T P^T) \left[\begin{array}{cc} I & 0 \\ 0 & \tilde{\Pi}^T \end{array}\right] \tilde{L} =$$

$$\left[\begin{array}{ccccc|cc}
J^0_{11} & J^0_{12} & & & & & \\
J^0_{21} & J^0_{22} & J^0_{23} & & & & \\
 & \ddots & \ddots & \ddots & & & \\
 & & & & J^0_{k-2,k-1} & & \\
 & & & J^0_{k-1,k-2} & J^0_{k-1,k-1} & J^0_{k-1,k} & \\
\hline
J^0_{k,1} & J^0_{k,2} & \dots & \dots & J^0_{k,k-1} & J^0_{k,k} & J^0_{k,k+1} \\
0 & 0 & \dots & \dots & 0 & J^0_{k+1,k} & J^0_{k+1,k+1}
\end{array}\right].$$

Each $J^0_{r,s}$ $(1 \le r, s \le k+1)$ denotes an $n \times n$ block, and they are defined as follows:

$$\begin{aligned}
J^0_{i,i} &= \left[\begin{array}{cc} (V_{i+1})_{11} & -(V_{i+1})_{12}\Delta^{22}_{i+1} \\ 0 & (V_i)_{22} \end{array}\right], \quad i = 1, \dots, k-1, \\
J^0_{i,i+1} &= \left[\begin{array}{cc} 0 & (V_{i+1})_{12} \\ 0 & 0 \end{array}\right], \quad i = 1, \dots, k-2, \\
J^0_{i+1,i} &= \left[\begin{array}{cc} 0 & 0 \\ (V_{i+1})_{21} & -(V_{i+1})_{22}\Delta^{22}_{i+1} \end{array}\right], \quad i = 1, \dots, k-2, \\
J^0_{k,i} &= \left[\begin{array}{cc} -(V_1)_{11}\Delta^{11}_1 S^{-1}_{i+1} & -(V_1)_{11}\Delta^{11}_1 S^{-1}_{i+1}\Delta^{12}_{i+1} \\ 0 & 0 \end{array}\right], \quad i = 1, \dots, k-2,
\end{aligned}$$

$$\begin{aligned}
J^0_{k,k-1} &= \begin{bmatrix} -(V_1)_{11}\Delta_1^{11}S_k^{-1} & -(V_1)_{11}\Delta_1^{11}S_k^{-1}\Delta_k^{12} \\ (V_k)_{21} & -(V_k)_{22}\Delta_k^{22} \end{bmatrix}, \\
J^0_{k-1,k} &= \begin{bmatrix} 0 & (V_k)_{12} \\ 0 & 0 \end{bmatrix} \tilde{\Pi}^T \tilde{\tilde{L}}, \\
\begin{bmatrix} J^0_{k,k} & J^0_{k,k+1} \\ J^0_{k+1,k} & J^0_{k+1,k+1} \end{bmatrix} &= \begin{bmatrix} (V_1)_{11} & & \\ & (V_k)_{22} & \\ & & I \end{bmatrix} \tilde{\Pi}^T \tilde{\tilde{L}}.
\end{aligned}$$

We now define a sequence of matrices J^i, $i = 0, 1, \ldots, k$ by

$$J^i = \bar{L}_i \ldots \bar{L}_1 \; J^0. \tag{32}$$

The following lemma summarizes some observations about each J^i:

LEMMA 4.4.

(i) $J^j_{i-2,i} = 0$, $j = 0, 1, \ldots, k+1$, $i = 3, 4, \ldots, k+1$;

(ii) $J^i_{i+1,i} = 0$, $i = 1, 2, \ldots, k$;

(iii) $J^j_{j+2,i} = J^0_{j+2,i}$, $j = 0, 1, \ldots, k-3$, $i = 0, 1, \ldots, k+1$.

(iv) $J^k_{j,i} = J^j_{j,i}$ *for* $j = 1, 2, \ldots, k-1$ *and* $i = 0, 1, \ldots, k+1$.

Proof. (i) follows from the fact that each of the "block elementary" matrices $\bar{L}_i$ is lower triangular, and so each J^i is block lower Hessenberg. For *(ii)*, note from (29) that J^k is block upper triangular, and so $J^k_{i+1,i} = 0$. Premultiplication by $\bar{L}_i$ affects only block rows i, $i+1$ and k of J^{i-1}, and no structural fill-in is created in block row i. It follows that $J^k_{i+1,i} = J^{k-1}_{i+1,i} = \ldots = J^i_{i+1,i} = 0$. Note for (iii) that transformations $\bar{L}_1, \ldots, \bar{L}_j$ affect only block rows $1, 2, \ldots, j+1$ and k of the target matrix J^0. In particular, block row $j+2$ is unaltered. Assertion (iv) follows from the fact that rows $1, \ldots, j$ are unaffected by the transformations $\bar{L}_{j+1}, \ldots, \bar{L}_k$. ∎

From Lemma 4.4 (ii), (iii), we have for $i = 1, \ldots, k-2$ that

$$\begin{aligned}
0 = J^i_{i+1,i} &= P_a \bar{L}_i^2 J^{i-1}_{i,i} + J^{i-1}_{i+1,i} \\
&= \begin{bmatrix} 0 & 0 \\ (\bar{L}_i^2)_{21} & (\bar{L}_i^2)_{22} \end{bmatrix} J^{i-1}_{i,i} + \begin{bmatrix} 0 & 0 \\ (V_{i+1})_{21} & -(V_{i+1})_{22}\Delta_{i+1}^{22} \end{bmatrix}.
\end{aligned} \tag{33}$$

For $i = 2, \ldots, k-2$,

$$\begin{aligned}
J^{i-1}_{i,i} &= P_a \bar{L}_{i-1}^2 J^{i-2}_{i-1,i} + J^{i-2}_{i,i} \\
&= \begin{bmatrix} 0 & 0 \\ (\bar{L}_{i-1}^2)_{21} & (\bar{L}_{i-1}^2)_{22} \end{bmatrix} J^{i-2}_{i-1,i} + \begin{bmatrix} (V_{i+1})_{11} & -(V_{i+1})_{12}\Delta_{i+1}^{22} \\ 0 & (V_i)_{22} \end{bmatrix}.
\end{aligned} \tag{34}$$

For $i = 3, \ldots, k-2$,

$$J^{i-2}_{i-1,i} = P_a \bar{L}_{i-2} J^{i-3}_{i-2,i} + J^{i-3}_{i-1,i}.$$

By Lemma 4.4 (i), $J^{i-3}_{i-2,i} = 0$, and so

$$J^{i-2}_{i-1,i} = J^{i-3}_{i-1,i} = \begin{bmatrix} 0 & (V_i)_{12} \\ 0 & 0 \end{bmatrix}. \tag{35}$$

By combining (35) and (34), we obtain

$$J^{i-1}_{i,i} = \begin{bmatrix} (V_{i+1})_{11} & -(V_{i+1})_{12}\Delta_{i+1}^{22} \\ 0 & (\bar{L}_{i-1}^2)_{21}(V_i)_{12} + (V_i)_{22} \end{bmatrix}.$$

Substituting this expression in (33), and looking at the $(2,1)$ and $(2,2)$ blocks of the product, we find that for $i = 3, \ldots, k-1$,

$$0 = (\bar{L}_i^2)_{21}(V_{i+1})_{11} + (V_{i+1})_{21} \tag{36}$$
$$0 = -\left[(V_{i+1})_{22} + (\bar{L}_i^2)_{21}(V_{i+1})_{12}\right]\Delta_{i+1}^{22} + (\bar{L}_i^2)_{22}\left[(V_i)_{22} + (\bar{L}_{i-1}^2)_{21}(V_i)_{12}\right]. \tag{37}$$

Consideration of the special cases $i = 1$ and $i = 2$ shows that (with the convention $\bar{L}_0 = I$) the formulae (36) and (37) hold in these cases as well. From (36),

$$\left[\ (\bar{L}_i^2)_{21} \mid I\ \right]\left[\begin{array}{cc}(V_{i+1})_{11} & (V_{i+1})_{12}\\ (V_{i+1})_{21} & (V_{i+1})_{22}\end{array}\right] = \left[\ 0 \mid (\bar{L}_i^2)_{21}(V_{i+1})_{12} + (V_{i+1})_{22}\ \right].$$

Since $\left[\ (\bar{L}_i^2)_{21} \mid I\ \right]$ has full rank and V_{i+1} is orthogonal, $(\bar{L}_i^2)_{21}(V_{i+1})_{12} + (V_{i+1})_{22}$ is nonsingular. Hence we can define the transformation

$$y_a^{i+1} = [(\bar{L}_i^2)_{21}(V_{i+1})_{12} + (V_{i+1})_{22}]^{-1}\bar{z}_a^{i+1}, \tag{38}$$

for $i = 1, \ldots, k-2$, with $y_a^1 = \bar{z}_a^1$. Substituting in (28) from (38), we have the following recurrence for $i = 2, \ldots, k$:

$$y_a^i = \Delta_i^{22}y_a^{i-1} + [(\bar{L}_{i-1}^2)_{21}(V_i)_{12} + (V_i)_{22}]^{-1}[(\bar{L}_{i-1}^2)_{21}z_b^i + z_a^i]. \tag{39}$$

Theorem 4.1 implies that this recurrence is stable. Examination of the other term in (28) leads to

$$\begin{aligned} {}^{(i-1)}z_b^1 &= (\bar{L}_{i-1}^2)_{12}\bar{z}_a^{i-1} + {}^{(i-2)}z_b^1 + (\bar{L}_{i-1}^2)_{11}z_b^{i+1} \\ &= \sum_{j=1}^{i-1}(\bar{L}_j^2)_{12}\bar{z}_a^j + z_b^1 + \sum_{j=1}^{i-1}(\bar{L}_j^2)_{11}z_b^{j+2}. \end{aligned}$$

Since the components of $\bar{L}_j^2$ are bounded in magnitude by 1, and since the recurrence underlying $\bar{z}_a^j$ is stable, there is no possibility of $\|{}^iz_b^1\|$ being exponentially larger than $\|z\|$.

Having shown that forward substitution involving the factor L is stable, we turn to the back substitution, which involves U.

From (29) and (32), we have that

$$J \overset{\text{def}}{=} J^k = U\tilde{D}\tilde{U}^{-1} \Leftrightarrow J\tilde{U} = U\tilde{D}. \tag{40}$$

Since J is both block lower Hessenberg and block upper triangular, it must be block upper bidiagonal. We wish to verify that the solution x of the system

$$Ux = \tilde{z} \tag{41}$$

does not have $\|x\|/\|\tilde{z}\|$ exponential in k. By an orthogonal change of variables $\tilde{D}^T x = \bar{x}$, (41) becomes

$$U\tilde{D}\bar{x} = \tilde{z} \Leftrightarrow J\tilde{U}\bar{x} = \tilde{z}. \tag{42}$$

We can, therefore, prove the desired result by showing that back substitution with both J and $\tilde{U}$ is stable.

For the case of $\tilde{U}$, this is best seen by partitioning the $(k+1)n$−vectors $\bar{x}$ and y in the usual way, namely,

$$\begin{aligned} \bar{x} &= (\bar{x}_b^1, \bar{x}_a^1, \bar{x}_b^2, \bar{x}_a^2, \ldots, \bar{x}_b^{k+1}, \bar{x}_a^{k+1}) \\ y &= (y_b^1, y_a^1, y_b^2, y_a^2, \ldots, y_b^{k+1}, y_a^{k+1}). \end{aligned}$$

To solve $\tilde{U}\bar{x} = y$, we first back substitute to find the solution of

$$\tilde{\tilde{U}}\left[\begin{array}{c}\bar{x}_b^k\\ \bar{x}_a^k\\ \bar{x}_b^{k+1}\\ \bar{x}_a^{k+1}\end{array}\right]$$

and then perform the following recurrence for $i = k-1, k-2, \ldots, 1$:

$$\tilde{U}_i \begin{bmatrix} \bar{x}_b^i \\ \bar{x}_a^i \end{bmatrix} + \tilde{E}_i \begin{bmatrix} \bar{x}_b^{i+1} \\ \bar{x}_a^{i+1} \end{bmatrix} + \tilde{G}_i \begin{bmatrix} \bar{x}_b^{k+1} \\ \bar{x}_a^{k+1} \end{bmatrix} = \begin{bmatrix} y_b^i \\ y_a^i \end{bmatrix}, \qquad i = k-1, k-2, \ldots, 1. \tag{43}$$

Componentwise, (43) is

$$\begin{aligned} -\Delta_{i+1}^{11} \bar{x}_b^i - \Delta_{i+1}^{12} \bar{x}_a^i + \bar{x}_b^{i+1} &= y_b^i \\ \bar{x}_a^i - T_i \bar{x}_a^{k+1} &= y_a^i. \end{aligned}$$

From Theorem 4.1, T_i decays exponentially with i, so the $\bar{x}_a^i$, $i = 1, 2, \ldots, k-1$, remain bounded. The homogeneous part of the first recurrence is

$$\bar{x}_b^i = (\Delta_{i+1}^{11})^{-1} \bar{x}_b^{i+1}.$$

Again, we appeal to Theorem 4.1 to deduce that this is stable.

For the back substitution with J, we again refer to the left-hand side of (29) to find expressions for $J_{i,i}$ and $J_{i,i+1}$. First, from Lemma 4.4 and the fact that each J^i is lower Hessenberg, we have for $i = 1, \ldots, k-1$, that

$$J_{i,i+1} = J_{i,i+1}^i = \bar{L}_i^1 J_{i,i+1}^{i-1} = \bar{L}_i^1 [P_a \bar{L}_{i-1}^2 J_{i-1,i+1}^{i-2} + J_{i,i+1}^{i-2}] = \bar{L}_i^1 \begin{bmatrix} 0 & (V_{i+1})_{12} \\ 0 & 0 \end{bmatrix}.$$

Second,

$$J_{i,i} = J_{i,i}^i = \bar{L}_i^1 J_{i,i}^{i-1} = \bar{L}_i^1 [P_a \bar{L}_{i-1}^2 J_{i-1,i}^{i-2} + J_{i,i}^{i-2}],$$

and since

$$J_{i-1,i}^{i-2} = P_a \bar{L}_{i-2}^2 J_{i-2,i}^{i-3} + J_{i-1,i}^{i-3} = \begin{bmatrix} 0 & (V_i)_{12} \\ 0 & 0 \end{bmatrix},$$

we obtain by substitution that

$$J_{i,i} = \bar{L}_i^1 \left\{ \begin{bmatrix} 0 & 0 \\ 0 & (\bar{L}_{i-1}^2)_{21} (V_i)_{12} \end{bmatrix} + \begin{bmatrix} (V_{i+1})_{11} & -(V_{i+1})_{12} \Delta_{i+1}^{22} \\ 0 & (V_i)_{22} \end{bmatrix} \right\}.$$

(This formula also holds in the special cases $i = 1$ and $i = 2$, with a simplified derivation.) Given the system $Jy = \tilde{z}$, partitioning in the usual way, and using (25), we obtain for $i = 1, \ldots, k-1$, that

$$\begin{aligned} J_{i,i} y^i + J_{i,i+1} y^{i+1} &= (\bar{L}_i^1)^{-1} \tilde{z}^i \\ \Rightarrow \begin{bmatrix} (V_{i+1})_{11} & -(V_{i+1})_{12} \Delta_{i+1}^{22} \\ 0 & (\bar{L}_{i-1}^2)_{21} (V_i)_{12} + (V_i)_{22} \end{bmatrix} \begin{bmatrix} y_b^i \\ y_a^i \end{bmatrix} + \begin{bmatrix} 0 & (V_{i+1})_{12} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_b^{i+1} \\ y_a^{i+1} \end{bmatrix} &= \begin{bmatrix} \bar{z}_b^i \\ \bar{z}_a^i \end{bmatrix}. \end{aligned}$$

It follows immediately that

$$y_a^i = [(\bar{L}_{i-1}^2)_{21} (V_i)_{12} + (V_i)_{22}]^{-1} \bar{z}_a^i,$$

exactly as in (38), and from (39) we know that the sequence of y_a^i, $i = 1, \ldots, k-1$ is governed by a stable recurrence. For the remaining components,

$$y_b^i = (V_{i+1})_{11}^{-1} [\bar{z}_b^i + (V_{i+1})_{12} \Delta_{i+1}^{22} y_a^i - (V_{i+1})_{12} y_a^{i+1}].$$

$(V_{i+1})_{11}$ is clearly nonsingular, since it is a diagonal block in the nonsingular upper triangular matrix J. Recall that for $i = 2, 3, \ldots, k$, $V_i = \bar{O}_{i-1} Q_i$, where $\bar{O}_{i-1}$ is a permutation matrix and Q_i is generated as in (17) and (18). It is reasonable to expect that the behavior of the sequence $\{Q_i\}$ is governed by the behavior of some solution to the underlying continuous Lyapunov equation (see, for example, [2, 9]). Therefore, it tends to depend on i/k rather than k. Since, by observation, the permutation matrices O^2_{i-1} are also independent of the discretization (they tend to change at indices that correspond to fixed points in the interval $[a,b]$), we can safely make the following assumption:

ASSUMPTION 3. *$\|(V_{i+1})^{-1}_{11}\|_2 \le K_3$, $i = 1,\dots,k$, where K_3 is independent of k.*

The end results of the discussion above can be stated as a theorem:

THEOREM 4.5. *If Assumptions 1, 2 and 3 hold, then the linear system (3) arising from (1)–(2) can be solved in a stable way by using the "serial" version of Algorithm* SLU.

Stability of the multipartition versions of SLU can be proved in a similar way. As an aside, we note that the "standard" error analysis for Gaussian elimination factorizations (which would take account of the structure of the coefficient matrix in (3), but not the dichotomy properties of the underlying ODE) actually gives tighter bounds as the number of partitions increases, because the upper bound on element growth during the factorization decreases. For the serial version, element growth of $O(2^{(k+1)n})$ could in principle occur in the last block column during elimination with row partial pivoting. For the "cyclic reduction" version, this bound is only $(2^{n\log k})$ (i.e., $O(k^n)$).

5. Extension to Problems with Multipoint Conditions or Algebraic Parameters. Finally, we describe the extensions of SLU to problems with parameters, or multipoint/integral side conditions. We have already observed that all variants of SLU are simply Gaussian elimination with row partial pivoting, applied to a reordered coefficient matrix; for example, "serial" SLU is obtained by applying row partial pivoting to (20). In this section, we show serial SLU orderings analogous to (20) for these more general problems.

One form of the algebraic-parameter problem [2, p. 322] is

$$\begin{gathered} y' = M(t,\lambda)y + q(t,\lambda), \qquad t \in [a,b], \quad y \in \mathbf{R}^n, \quad \lambda \in \mathbf{R}^r \\ B_a y(a,\lambda) + B_b y(b,\lambda) = d(\lambda), \quad d \in \mathbf{R}^{(n+r)\times n}. \end{gathered}$$

If this problem is linearized with respect to λ, and a finite-difference or multiple shooting method is applied, the following extended form of (3) is obtained:

$$\begin{bmatrix} B_a & & & & B_b & Z_0 \\ A_1 & C_1 & & & & Z_1 \\ & A_2 & C_2 & & & Z_2 \\ & & \ddots & \ddots & & \vdots \\ & & & A_k & C_k & Z_k \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_{k+1} \\ \delta\lambda \end{bmatrix} = \begin{bmatrix} d(\lambda) \\ f_1 \\ f_2 \\ \vdots \\ f_k \end{bmatrix}. \tag{44}$$

Rearranging rows and columns, we obtain

$$A = \left[\begin{array}{ccccc|ccc} C_1 & & & & & & A_1 & Z_1 \\ A_2 & C_2 & & & & & 0 & Z_2 \\ & A_3 & C_3 & & & & 0 & Z_3 \\ & & \ddots & \ddots & & & \vdots & \vdots \\ & & & & C_{k-1} & & 0 & Z_{k-1} \\ \hline & & & & A_k & C_k & 0 & Z_k \\ & & & & & B_b & B_a & Z_0 \end{array}\right].$$

We can now proceed with SLU exactly as described in Section 2. The major difference is that the G_i blocks will now have $n+r$ columns, rather than n.

The multipoint side condition problem (see [2, pp. 6,323]) is

$$y' = M(t)y + q(t), \qquad t \in [a,b], \qquad y \in \mathbf{R}^n,$$

$$\begin{array}{l} \sum_{j=1}^{J} B_j y(\xi_j) = d, \quad d \in \mathbf{R}^n, \\ a = \xi_1 < \xi_2 < \ldots < \xi_J = b. \end{array}$$

When the points $\xi_1, \ldots, \xi_J$ are included in the finite difference or multiple shooting mesh, the generalization of (3), with some trivial row reordering, is

$$\begin{bmatrix} A_1 & C_1 & & & & & & \\ & A_2 & C_2 & & & & & \\ & & \ddots & \ddots & & & & \\ & & & & A_j & C_j & & \\ & & & & & \ddots & \ddots & \\ & & & & & & A_k & C_k \\ B_1 & 0 & \ldots & 0 & B_j & 0 & \ldots & B_J \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_k \\ d \end{bmatrix}. \tag{45}$$

Taking the transpose of the coefficient matrix, we obtain

$$A^T = \begin{bmatrix} A_1^T & & & & & & B_1^T \\ C_1^T & A_2^T & & & & & 0 \\ & C_2^T & \ddots & & & & \vdots \\ & & \ddots & A_j^T & & & B_j^T \\ & & & C_j^T & \ddots & & 0 \\ & & & & \ddots & A_k^T & \vdots \\ & & & & & C_k^T & B_J^T \end{bmatrix}. \tag{46}$$

Clearly, (46) differs from (20) only in that the last block column contains some extra nonzeros. Since `SLU` fills out this last column in any case, this is of little concern. Hence, (45) can be solved by applying `SLU` to A^T rather than A. `SLU` computes a factorization of the form

$$PA^TQ = LU,$$

where P and Q are permutation matrices. Since

$$As = f \Leftrightarrow U^T L^T (Ps) = Q^T f,$$

the solution of (45) is obtained by performing a forward substitution with U^T and a back substitution with L^T.

If the multipoint condition in (45) is replaced by an integral condition of the form

$$\int_a^b B(\tau) y(\tau)\, d\tau = d, \tag{47}$$

then appropriate discretization of (47) will produce a problem of the form (45), with $J = k+1$. The resulting technique provides an alternative to the popular approach of introducing artificial dependent variables to obtain a new problem with boundary conditions only.

REFERENCES

[1] U. M. Ascher and P. S. Y. Chan, *On parallel methods for boundary value odes*, Computing, 46 (1991), pp. 1–17.

[2] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Prentice-Hall, Englewood Cliffs, 1988.

[3] D. L. Brown and J. Lorenz, *A high order method for stiff boundary value problems with turning points*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), pp. 790–805.

[4] L. Dieci, M. R. Osborne, and R. D. Russell, *A Riccati transformation method for solving linear BVPs. I: Theoretical aspects*, SIAM Journal on Numerical Analysis, 25 (1988), pp. 1055–1073.

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

[6] H. B. Keller and M. Lentini, *Invariant imbedding, the box scheme and an equivalence between them*, SIAM Journal on Numerical Analysis, 19 (1982), pp. 942–962.

[7] M. Lentini and V. Pereyra, *An adaptive finite difference solver for nonlinear two-point boundary value problems with mild boundary layers*, SIAM Journal on Numerical Analysis, 14 (1977), pp. 91–111.

[8] R. M. M. Mattheij, *Stability of block LU-decompositions of matrices arising from BVP*, SIAM Journal on Algebraic and Discrete Methods, 5 (1984), pp. 314–331.

[9] ———, *Decoupling and stability of algorithms for boundary value problems*, SIAM Review, 27 (1985), pp. 1–44.

[10] J. M. Varah, *Alternate row and column elimination for solving certain linear systems*, SIAM Journal on Numerical Analysis, 13 (1976), pp. 71–75.

[11] S. J. Wright, *Stable parallel algorithms for two-point boundary value problems*, Tech. Rep. MCS–P178–0990, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, September 1990.