Parallel stable compactification for ODE with parameters and multipoint conditions

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computing Science

RANA 92-11
September 1992

PARALLEL STABLE COMPACTIFICATION FOR ODE WITH PARAMETERS AND MULTIPOINT CONDITIONS

by

R.M.M. Mattheij
S.J. Wright

Reports on Applied and Numerical Analysis
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven
The Netherlands
ISSN: 0926–4507

PARALLEL STABLE COMPACTIFICATION FOR ODE WITH PARAMETERS AND MULTIPOINT CONDITIONS*

R. M. M. MATTHEIJ† AND S. J. WRIGHT‡

Abstract. Many algorithms for solving ordinary differential equations with parameters and multipoint side conditions give rise to systems of linear algebraic equations in which the coefficient matrices have a bordered block diagonal structure. In this paper, we show how these problems can be solved by using parallel algorithms based on stabilized compactification.

Key words. ordinary differential equations, two-point boundary value problems, multipoint conditions, parallel computing

1. Introduction.

We discuss stable algorithms for solving algebraic linear systems arising from ordinary differential and difference equations. These equations may include some global parameters, and their solutions are subject to multipoint side conditions. To make the discussion in subsequent sections clearer, we group the problems to be addressed into three categories:

Two-point boundary value problems:

$$\begin{aligned} \dot{x} &= A(t)x + c(t), \qquad t \in [a,b], \\ M_a x(a) + M_b x(b) &= d, \qquad x(t),\ c(t),\ d \in \mathbb{R}^n; \end{aligned} \tag{1.1}$$

Parametrized problems with multipoint conditions:

$$\begin{aligned} \dot{x} &= A(t)x + C(t)\lambda + c(t), \qquad t \in [a,b], \\ \textstyle\sum_{j=1}^{p} M_i x(\tau_j) + N\lambda &= d, \qquad a \le \tau_1 \le \cdots \le \tau_p \le b, \\ x(t),\ c(t) \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}^m, &\quad M_i \in \mathbb{R}^{(m+n)\times n}, \quad N \in \mathbb{R}^{(m+n)\times m}; \end{aligned} \tag{1.2}$$

Parameter identification problems (see, for example, Bock [3]):

$$\begin{aligned} \min \textstyle\sum_{j=1}^{p} \|M_j x(\tau_j) + N_j\lambda - d_j\|^2 + \|N_0\lambda - d_0\|^2, \\ \dot{x} = A(t)x + C(t)\lambda + b(t), \qquad t \in [a,b] \\ x(t) \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^m, \quad M_j \in \mathbb{R}^{n_j\times n},\ N_j \in \mathbb{R}^{n_j \times m}. \end{aligned} \tag{1.3}$$

Although these categories appear to be listed in order of increasing generality, the ordering is not strict: problems of the form (1.2) can be recast in the form (1.1) (see, for example, Ascher and Russell [2]). For purposes of efficiency, it is usually best to use the original form when solving such problems.

* 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.

† Department of Mathematics and Computing Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands.

‡ Argonne National Laboratory, 9700 South Cass Avenue, Argonne, Illinois 60439, USA.

Each of these problem categories has a corresponding "discrete" form, in which the differential equation is replaced by a difference equation. We categorize the discrete problems similarly:

$$\begin{aligned} A_i x_i + B_i x_{i+1} &= c_i, \qquad i = 1, \cdots, N, \\ M_a x_1 + M_b x_{N+1} &= d, \qquad x_i,\ c_i,\ d \in \mathbb{R}^n; \end{aligned} \tag{1.4}$$

$$\begin{gathered} A_i x_i + B_i x_{i+1} + C_i \lambda = c_i, \qquad i = 1, \cdots, N, \\ \textstyle\sum_{i=1}^{N+1} M_i x_i + N\lambda = d, \\ x_i,\ c_i \in \mathbb{R}^n, \qquad \lambda \in \mathbb{R}^m, \qquad M_i \in \mathbb{R}^{(m+n)\times n}, \qquad N \in \mathbb{R}^{(m+n)\times m}; \end{gathered} \tag{1.5}$$

$$\begin{gathered} \min \textstyle\sum_{i=1}^{N+1} \|M_i x_i + N_i \lambda - d_i\|^2 + \|N_0 \lambda - d_0\|^2, \\ A_i x_i + B_i x_{i+1} + C_i \lambda = c_i, \qquad i = 1, \cdots, N, \\ x_i \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^m, \qquad M_i \in \mathbb{R}^{n_i \times n},\ N_i \in \mathbb{R}^{n_i \times m}. \end{gathered} \tag{1.6}$$

The "discrete" categories can be obtained by applying algorithms such as multiple shooting or finite differencing to the corresponding "continuous" categories. The widely used collocation algorithm also gives rise to such discrete problems after a condensation step has been applied.

We will not address the special difficulties caused by stiffness and singularly perturbed problems. Neither will we specifically address nonlinear problems, though it is well known that algorithms for such problems often require the solution of linear systems like those described above as a core operation.

In the next section, we will review the concept of conditioning for the continuous problems above, and its implications for the conditioning of the linear systems to be solved in the discrete problems (1.4), (1.5), and (1.6). In §3, a parallel algorithm for (1.4) is described and is contrasted with previously proposed algorithms (e.g., Wright [12, 11]). This algorithm is extended to problems of the form (1.5) and (1.6) in §5. Finally, some numerical results are presented in §6.

2. Conditioning of the Problem and Structure of the Solution Spaces.

The various problems defined in §1 relate to solution spaces of potentially different types. This variety of solution types influences the possibilities for parallelization that will be discussed in the next section. Here we characterize these solution spaces; that is, we describe certain growth properties of homogeneous modes in the fundamental solution $\Phi(t)$. Throughout, we assume that the problem is *well conditioned*. By this we mean that small perturbations of inhomogeneities in the ODE and the side conditions manifest themselves in the solution as quantities that are only a moderate constant larger in norm than the original perturbations, that is,

$$\|\delta x\| \le \kappa \max(|d|, \|c\|), \tag{2.1}$$

where $\|.\|$ is a suitable function space norm, in either the continuous or discrete setting, and $|.|$ is any Hölder norm.

In the case of the two-point boundary value problem (1.1), well-conditioning implies that the underlying solution space is *dichotomic*; that is, there exists a projection P and a moderate constant $\tilde{\kappa}$ such that

$$\text{(2.2)} \qquad \begin{cases} |\Phi(t)P\Phi^{-1}(s)| \leq \tilde{\kappa}, & t > s, \\ |\Phi(t)(I-P)\Phi^{-1}(s)| \leq \tilde{\kappa}, & t < s. \end{cases}$$

(See de Hoog and Mattheij [5].) Hence, Φ can be properly split into one set of solution modes that do not (significantly) increase and another set of modes that do not (significantly) decrease.

If we have multipoint side conditions ($m = 0$ in (1.2)), then well-conditioning allows for a "switch" of growth behavior at any of the internal conditions. This property was referred to by de Hoog and Mattheij [5] as *polychotomy*. Specifically, we can choose projections $P_1, \cdots, P_M$ ($M \leq \min(p,n)$) such that $\sum_{j=1}^{M} P_j = I$, $P_i P_j = P_j P_i = \delta_{ij} P_j$, and

$$\text{(2.3)} \qquad \begin{cases} |\Phi(t)\sum_{j=1}^{k} P_j \Phi^{-1}(s)| \leq \tilde{\kappa}, & \beta_k < s \leq \beta_{k+1}, & t > s, \\ |\Phi(t)\sum_{j=k+1}^{M} P_j \Phi^{-1}(s)| \leq \tilde{\kappa}, & \beta_k < s \leq \beta_{k+1}, & t < s, \end{cases}$$

where, again, $\tilde{\kappa}$ is a moderate constant and the switching points $a = \beta_1, \beta_2, \cdots, \beta_{M+1} = b$ are a subset of $\{a, \tau_1, \cdots, \tau_p, b\}$. Polychotomy means that Φ is dichotomic on each subinterval (β_k, β_{k+1}) but that the dimension of the subspace of "nonincreasing" modes may become larger — by rank(P_k) — at each switching point β_k.

A more or less complementary result to the foregoing holds for "pure" parameter problems (for which $p = 2$, $a = \tau_1$ and $b = \tau_2$ in (1.2)). Again, given well-conditioning, Mattheij [10] showed that there exist a moderate constant $\tilde{\kappa}$, projections $P_1, \cdots, P_M$ with ($M \leq \min(m,n)$) satisfying the conditions described above, and a set of switching points α_k, $k = 1, \cdots M+1$ with $a = \alpha_1 < \alpha_2 < \cdots < \alpha_{M+1} = b$ such that

$$\text{(2.4)} \qquad \begin{cases} |\Phi(t)\sum_{j=k+1}^{M} P_j \Phi^{-1}(s)| & \leq & \tilde{\kappa}, & \alpha_k < s \leq \alpha_{k+1}, & t > s, \\ |\Phi(t)\sum_{j=1}^{k} P_j \Phi^{-1}(s)| & \leq & \tilde{\kappa}, & \alpha_k < s \leq \alpha_{k+1}, & t < s \end{cases}$$

Again, Φ is dichotomic on each subinterval, but the dimension of the subspace of nonincreasing modes may now become *smaller* at each switchpoint α_k as we move from left to right.

In the more general case of $m \neq 0$ and $p > 2$ in (1.2), one should expect that any combination of (2.3) and (2.4) can occur. We will use the term "polychotomy" to refer to the multiple splitting of the solution space in this case as well.

Thus, we may expect for (1.2) a total of at most $\min(p,n)$ switching points at which the dimension of the subspace of nonincreasing modes may increase, and at most $\min(m,n)$ switching points at which this dimension may decrease. Note, however, that a typical "pure" mode of this subspace may undergo only one switch in behavior; otherwise, it would violate the local well-conditioning on each subinterval (β_k, β_{k+1}) or (α_k, α_{k+1}).

It should be noted that the potential switching points τ_k (from which the β_k are chosen) are known beforehand, while the α_k may be anywhere in the interval (a,b).

3. The Basic Algorithm. The "stabilized compactification" algorithm has been described in Ascher, Mattheij, and Russell [1, pp. 157–161]. In this section, we outline a parallel version and discuss some aspects of its implementation in a message-passing computational environment. We describe the method for the case of problem (1.4); this is the simplest of our discrete problems since, when it is well conditioned, the dimensions of the nonincreasing and nondecreasing subspaces remain constant across all the stages. We use ℓ to denote the dimension of the nonincreasing subspace. The dimension of the complementary nondecreasing subspace will therefore be $(n-\ell)$. For the moment, we also assume that the value of ℓ is known.

Suppose that the $N+1$ indices in the problem (1.4) are split into p partitions of approximately equal size. This is done by choosing indices $k_1, k_2, k_3, \cdots, k_{p+1}$ such that

$$\begin{aligned} k_1 &= 0, & k_{p+1} &= N, \\ k_{i+1} &\ge k_i + 2, & i &= 1, \cdots, p. \end{aligned}$$

The j-th partition of the linear system is taken to be

$$\text{(3.1)} \quad \begin{bmatrix} A_{k_j+1} & B_{k_j+1} & & \\ & A_{k_j+2} & B_{k_j+2} & \\ & & \ddots & \ddots \\ & & A_{k_{j+1}} & B_{k_{j+1}} \end{bmatrix} \begin{bmatrix} x_{k_j+1} \\ x_{k_j+2} \\ \vdots \\ x_{k_{j+1}+1} \end{bmatrix} = \begin{bmatrix} b_{k_j+1} \\ b_{k_j+2} \\ \vdots \\ b_{k_{j+1}} \end{bmatrix}.$$

The algorithm starts by transforming the recurrence (3.1) to one involving upper triangular blocks. Then, explicit decoupling of nonincreasing and nondecreasing modes is used to find particular and fundamental solutions for this partition. Because of the decoupling, element growth in the fundamental solution is avoided.

Suppose for the moment that we can choose an $n \times n$ orthogonal matrix $R^{(j)}_{k_j+1}$ that is effective in decoupling these two sets of modes. (The choice of $R^{(j)}_{k_j+1}$ will be discussed further below.) Starting with this matrix, the transformation process performs repeated QR (orthogonal) factorizations to find the following matrices, all of which are square with dimension n:

$$\begin{array}{lll} R_i^{(j)}, & i = k_j+2, \cdots, k_{j+1}, & \text{orthogonal,} \\ Q_i^{(j)}, & i = k_j+1, \cdots, k_{j+1}-1, & \text{orthogonal,} \\ U_i^{(j)}, & i = k_j+1, \cdots, k_{j+1}-1, & \text{upper triangular,} \\ V_i^{(j)}, & i = k_j+1, \cdots, k_{j+1}-1, & \text{upper triangular,} \end{array}$$

such that

$$\text{(3.2a)} \quad A_i R_i^{(j)} = Q_i^{(j)} U_i$$

$$\text{(3.2b)} \quad Q_i^{(j)T} B_i = V_i R_{i+1}^{(j)T}.$$

The recurrence

$$A_i x_i + B_i x_{i+1} = c_i, \qquad i = k_j+1, \cdots, k_{j+1}-1,$$

has now been transformed to

$$\text{(3.3)} \quad U_i z_i + V_i z_{i+1} = Q_i^T c_i, \qquad i = k_j+1, \cdots, k_{j+1}-1,$$

where

$$z_i = R_i^T x_i.$$

(Here and subsequently, the superscripts on $R_i^{(j)}$, $Q_i^{(j)}$, etc., will be dropped when their values are clear from the context.) ¿From an implementation point of view, these operations create little fill-in. The matrices U_i and V_i can be stored in the upper triangles of the data structures formerly occupied by A_i and B_i. Most of the information needed to reconstruct the orthogonal matrices Q_i and R_i can be stored in the lower triangles, though an extra n-vector is also needed for each matrix.

We now construct a fundamental solution to (3.3), denoted by

$$\{\Phi_i\}_{k_j+1}^{i=k_j+1}, \qquad \Phi_i \in \mathbb{R}^{n\times n}, \qquad \text{upper triangular,} \tag{3.4}$$

that satisfies the boundary conditions

$$(\Phi_{k_j+1})_{2,.} = [0 \mid I], \qquad (\Phi_{k_{j+1}})_{1,.} = [I \mid 0]. \tag{3.5}$$

Here, $(.)_{1,.}$ denotes the first $(n-\ell)$ rows of a matrix and $(.)_{2,.}$ denotes the last ℓ rows. The remaining components of these two sequences are computed by back substitution with the matrix formed by the recurrence (3.3). The $(.)_{2,.}$ components are calculated in a forward sweep and the $(.)_{1,.}$ components in a reverse sweep. Specifically,

for $i = k_j + 1, \cdots, k_{j+1} - 1$

$$(\Phi_{i+1})_{2,.} = -(V_i)_{22}^{-1}[0 \mid (U_i)_{22}(\Phi_i)_{22}]; \tag{3.6}$$

for $i = k_{j+1} - 1, \cdots, k_j + 1$

$$(\Phi_i)_{1,.} = -(U_i)_{11}^{-1}\left\{(V_i)_{11}(\Phi_{i+1})_{1,.} + (V_i)_{12}(\Phi_{i+1})_{2,.} + (U_i)_{12}(\Phi_i)_{2,.}\right\}. \tag{3.7}$$

Here, $(.)_{11}$ is the principal $(n-\ell)\times(n-\ell)$ submatrix of an $n \times n$ matrix; the three blocks $(.)_{22}$, $(.)_{12}$, and $(.)_{21}$ complete a 2×2 block partitioning. Examination of (3.6) and (3.7) will confirm that the $\Phi_i^{(j)}$ are upper triangular. The homogeneous parts of (3.6) and (3.7) have coefficients $(V_i)_{22}^{-1}(U_i)_{22}$ and $-(U_i)_{11}^{-1}(V_i)_{11}$, respectively. By our original assumption of well-conditioning, the recurrences are stable; they are effectively producing components of the nonincreasing modes (forward sweep) and nondecreasing modes (backward sweep), provided that the initial matrix R_{k_j} has been chosen in a suitable way.

The particular solution, denoted by

$$\{\hat{z}_i\}_{i=k_j+1}^{k_{j+1}},$$

is formed in a similar way. The boundary conditions are

$$(\hat{z}_{k_j+1})_{2,.} = 0, \qquad (\hat{z}_{k_{j+1}})_{1,.} = 0. \tag{3.8}$$

If one uses the notation $\hat{c}_i = Q_i^T c_i$, the forward and reverse sweeps have the form

for $i = k_j + 1, \cdots, k_{j+1} - 1$

$$(\hat{z}_{i+1})_2 = (V_i)_{22}^{-1}\left\{(\hat{c}_{i+1})_2 - (U_i)_{22}(\hat{z}_i)_2\right\}; \tag{3.9}$$

for $i = k_{j+1} - 1, \cdots, k_j + 1$

$$(3.10)\quad (\hat{z}_i)_1 = (U_i)_{11}^{-1} \{(\hat{c}_i)_1 - (V_i)_{11}(\hat{z}_{i+1})_1 - (V_i)_{12}(\hat{z}_{i+1})_2 - (U_i)_{12}(\hat{z}_i)_2\}.$$

We now deal with the issue of choosing the values of ℓ and R_{k_j+1}. Mattheij [9, p. 329] describes a heuristic that, applied to the present situation, would find R_{k_j+1} so that the diagonal elements of $V_i^{-1}U_i$ (namely, $(U_i)_{ll}/(V_i)_{ll}$, $l = 1, \cdots, n$) appear in descending order for most $i = k_j + 1, \cdots, k_{j+1} - 1$. We generalize this heuristic slightly for two reasons. First, we usually do not have boundary condition information to guide the choice of R_{k_j+1}; and second, we wish to allow possible singularity of U_i and V_i (that is, of A_i and B_i.)

To make an initial guess at R_{k_j+1}, we make use of the generalized singular value decomposition. In the following result, which is a direct consequence of Theorem 8.7.4 of Golub and Van Loan [6], all matrices are assumed to be $n \times n$.

THEOREM 3.1. *Provided that $[A, B]$ has full rank n, there are orthogonal matrices P, Q, and W and an upper triangular matrix T such that*

$$W^T T^{-T} AP = C, \qquad W^T T^{-T} BQ = S,$$

where $C = \text{diag}(c_1, \cdots, c_n)$ *and* $S = \text{diag}(s_1, \cdots, s_n)$, *with* $1 \geq c_1 \geq \cdots \geq c_n \geq 0$ *and* $0 \leq s_1 \leq \cdots \leq s_n \leq 1$, *and* $c_i^2 + s_i^2 = 1$, $i = 1, \cdots, n$.

Our initial guess for R_{k_j+1} is obtained as follows:

(i) Set $A = A_{k_j+1}$, $B = B_{k_j+1}$, and compute W, T, P, Q, C, and S as in Theorem 3.1.

(ii) Perform a QR factorization to obtain Z ($n \times n$ orthogonal) and Y ($n \times n$ upper triangular) such that $T^T W = ZY$.

(iii) Set $R_{k_j+1} \leftarrow P$, $R_{k_j+2} \leftarrow Q$, $Q_{k_j+1} \leftarrow Z$, $U_{k_j+1} \leftarrow YC$, $V_{k_j+1} \leftarrow YS$.

We note that this construction is valid since, by our assumption that (1.4) is well conditioned, $[A_{k_j+1}, B_{k_j+1}]$ has full rank. It is also easy to check that equations (3.2) hold for $i = k_j + 1$. Moreover the diagonals of U_{k_j+1} and V_{k_j+1} are ordered properly, relative to each other. By this, we mean that

- any zero diagonal elements in V_{k_j+1} occur in the upper left of the matrix;
- any zero diagonals in U_{k_j+1} occur in the lower right; and
- for the indices l for which both $(V_{k_j+1})_{ll}$ and $(U_{k_j+1})_{ll}$ are nonzero, the ratio

$$(U_{k_j+1})_{ll}/(V_{k_j+1})_{ll} = c_l/s_l$$

decreases as l increases.

Using this initial choice of R_{k_j+1}, we now compute a few more successive Q_i, R_i, U_i, and V_i by using the formulae (3.2) and check to see whether the diagonals of U_i and V_i continue to appear in the proper order. If not, the columns of R_{k_j+1} are permuted in accordance with the present ordering, and the orthogonal transformation process is begun anew.

Before computing the fundamental and particular solutions, we need to determine the dimension ℓ of the nonincreasing subspace. When the boundary conditions are *separated* (that is, zero rows of M_a correspond to nonzero rows of M_b, and vice versa) and the problem is well conditioned, ℓ may just be taken to be the number of nonzero rows in M_a. Otherwise, we require the lower right $\ell \times \ell$ submatrix $(V_i)_{22}$ of V_i to dominate the corresponding submatrix $(U_i)_{22}$ of U_i. For "close calls," we can use the test suggested in [9]: If the quantity

$$\sum_{j=1}^{p}\sum_{i=k_j+1}^{k_{j+1}-1}\log\left[\left(U_i^{(j)}\right)_{ll}/\left(V_i^{(j)}\right)_{ll}\right]$$

is negative, the l-th diagonal element is included in the $(2,2)$ partition. In a multiprocessor setting, the formation of this "global" sum requires interprocesssor communication. If vendor-supplied primitives for this type of operation are not available, it can be implemented easily by making a binary tree out the processors.

In partition j, each component x_i of the true solution can be expressed in terms of the fundamental and particular solutions in the following way:

$$z_i = \Phi_i^{(j)}s_j + \hat{z}_i^{(j)}, \qquad x_i = R_i^{(j)}z_i, \qquad i = k_j+1,\cdots,k_{j+1}, \tag{3.11}$$

where the values of $s_j \in \mathbb{R}^n$, $j = 1,\cdots,p$ and x_{N+1} are determined by solving a reduced system. The boundary conditions contribute one block row to this system:

$$\begin{aligned} &M_a x_1 + M_b x_{N+1} = d \\ &\Longrightarrow \quad M_a R_1^{(1)}(\Phi_1^{(1)}s_1 + \hat{z}_1^{(1)}) + M_b x_{N+1} = d. \end{aligned} \tag{3.12}$$

The remaining p blocks of the reduced system are obtained by considering the equations that bind adjacent partitions together. These are obtained by taking the last block equation from each partition. For $j = 2,\cdots,p$, we have

$$\begin{aligned} &A_{k_j}x_{k_j} + B_{k_j}x_{k_j+1} = c_{k_j} \\ &\Longrightarrow \quad A_{k_j}R_{k_j}^{(j-1)}\left[\Phi_{k_j}^{(j-1)}s_{j-1} + \hat{z}_{k_j}^{(j-1)}\right] + B_{k_j}R_{k_j+1}^{(j)}\left[\Phi_{k_j+1}^{(j)}s_j + \hat{z}_{k_j+1}^{(j)}\right] = c_{k_j}. \end{aligned} \tag{3.13}$$

The remaining equation, obtained from the last partition, is

$$A_N R_N^{(p)}\left[\Phi_N^{(p)}s_p + \hat{z}_N^{(p)}\right] + B_N x_{N+1} = c_N. \tag{3.14}$$

Aggregating (3.12), (3.13), and (3.14), and defining $s_{p+1} \overset{\Delta}{=} x_{N+1}$, we see that the reduced system has the form

$$\tilde{A}_j s_j + \tilde{B}_j s_{j+1} = \tilde{c}_j, \qquad j = 1,\cdots,p, \tag{3.15a}$$

$$\tilde{M}_a s_1 + M_b s_{p+1} = \tilde{d}. \tag{3.15b}$$

This system has the same form as the original system (1.4), which immediately suggests that the partitioning/reduction process can be performed *recursively*. Such an approach is indeed desirable when the number of processors is large, making the reduced system (3.15) too large to solve on a single processor. We return to this point in §6.

Tables 3.1 and 3.2 compare storage requirements and operation counts for five algorithms, including the one discussed above (partitioned stabilized compactification, or PSC). In reporting the operation counts, lower-order (n^2) terms are ignored. We assume that, for each algorithm, the calculation of fundamental and particular solutions was carried out in separate phases. It is therefore necessary to store enough information to reconstruct the transformations that were used to factor the matrix associated with the recurrence (1.4). The statistics for the structured QR algorithm

Operation counts and storage requirements for five algorithms, assuming separated end conditions ($N =$ number of stages, $n =$ dimension of each x_i, $\ell =$ number of left-hand end conditions, $R =$ number of right-hand sides, $p =$ number of partitions in first level of PSC*)*

Algorithm	Operation Count	Storage
LU (row pivoting)	$N[\frac{5}{3}n^3 + 3\ell n^2 + R(4n^2 + 2\ell n)]$	$3Nn^2$
DECOMP/SOLVE	$N[\frac{2}{3}n^3 + (4R + 5\ell)n^2 - 2n\ell^2]$	$2Nn^2$
Structured QR	$N[\frac{46}{3}n^3 + 11Rn^2]$	$4Nn^2$
Structured LU	$N[\frac{23}{3}n^3 + 8Rn^2]$	$4Nn^2$
PSC	$N[\frac{22}{3}n^3 + 6Rn^2] + p[\frac{46}{3}n^3 + 11Rn^2]$	$\frac{5}{2}Nn^2 + 4pn^2$

Operation counts and storage requirements for five algorithms, assuming coupled end conditions ($N =$ number of stages, $n =$ dimension of each x_i, $R =$ number of right-hand sides, $p =$ number of partitions in first level of PSC*)*

Algorithm	Operation Count	Storage
LU (row pivoting)	$N[\frac{23}{3}n^3 + 8Rn^2]$	$4Nn^2$
DECOMP/SOLVE	$N[\frac{14}{3}n^3 + 4Rn^2]$	$3Nn^2$
Structured QR	$N[\frac{46}{3}n^3 + 11Rn^2]$	$4Nn^2$
Structured LU	$N[\frac{23}{3}n^3 + 8Rn^2]$	$4Nn^2$
PSC	$N[\frac{22}{3}n^3 + 6Rn^2] + p[\frac{46}{3}n^3 + 11Rn^2]$	$\frac{5}{2}Nn^2 + 4pn^2$

differ slightly from those reported in Wright [12], since we refer here to an improved variant of the technique which is based on Givens rotations rather than Householder transformations. (This variant will be described in a forthcoming report.) The statistics for PSC assume that the reduced system (3.15) is solved by using structured QR. We use this approach in our implementation in §6, since it can be implemented stably, in a fashion akin to cyclic reduction, on a binary tree of processors.

For problems with separated end conditions, all five algorithms are stable, while only structured QR and PSC are guaranteed to be stable when the end conditions are coupled (although LU and structured LU are almost always stable in these circumstances). The LU and DECOMP/SOLVE algorithms are not parallelizable unless n is large.

As well as halving the computational cost of structured QR, PSC has a clear advantage in storage requirement. The Householder vectors that define the orthogonal matrices Q_i and R_i can be largely stored in the lower triangle that is vacated when A_i and B_i are transformed to U_i and V_i. The only real fill-in is due to the fundamental solution matrices Φ_i. When N is significantly larger than p, the factorization process requires only about 25% more storage than the cost of storing the original system.

The algorithm PSC can be viewed as a particular factorization of the matrix corresponding to the recurrence (1.4). We state this result as a theorem.

THEOREM 3.2. *Algorithm* PSC *is equivalent to a particular solution scheme for the linear system $Ax = c$, where*

$$A = \begin{bmatrix} A_1 & B_1 & & & \\ & A_2 & B_2 & & \\ & & \ddots & \ddots & \\ & & & A_N & B_N \\ M_a & & & & M_b \end{bmatrix}, \qquad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \\ d \end{bmatrix}.$$

In this scheme, A is factored in the form

$$P_L Q A R P_R = \begin{bmatrix} \hat{L}_1 & 0 \\ \hat{L}_2 & \tilde{A} \end{bmatrix} \begin{bmatrix} I & \hat{U}_2 \\ 0 & I \end{bmatrix} = \tilde{L}\tilde{U}, \tag{3.16}$$

where P_L and P_R are permutation matrices, Q and R are block diagonal matrices with orthogonal $n \times n$ blocks, and $\hat{L}_1$ is lower triangular. The block $\tilde{A}$ in (3.16) has the form

$$\tilde{A} = \begin{bmatrix} \tilde{A}_1 & \tilde{B}_1 & & & \\ & \tilde{A}_2 & \tilde{B}_2 & & \\ & & \ddots & \ddots & \\ & & & \tilde{A}_p & \tilde{B}_p \\ \tilde{M}_a & & & & M_b \end{bmatrix}$$

(that is, the coeficient matrix associated with the recurrence (3.15)). The remaining blocks in (3.16) are defined in the proof.

Proof. We prove only the case of $p > 1$ (trivial modifications are required for $p = 1$). The matrices Q and R come from the initial orthogonal transformation phase. They are defined as

$$\begin{aligned} Q^T &= \operatorname{diag}(Q_1^{(1)}, Q_2^{(1)}, \cdots, Q_{k_2-1}^{(1)}, I, Q_{k_2+1}^{(2)}, \cdots, Q_{k_2-1}^{(2)}, I, \cdots, Q_{k_{p+1}-1}^{(p)}, I, I), \\ R &= \operatorname{diag}(R_1^{(1)}, R_2^{(1)}, \cdots, R_{k_2}^{(1)}, R_{k_2+1}^{(2)}, \cdots, R_{k_3}^{(2)}, R_{k_3+1}^{(3)}, \cdots, R_{k_{p+1}}^{(p)}, I). \end{aligned}$$

Then QAR is

$$\begin{bmatrix} U_1 & V_1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & U_{k_2-1} & V_{k_2-1} & & & & \\ & & & (A_{k_2} R_{k_2}^{(1)}) & (B_{k_2} R_{k_2+1}^{(2)}) & & & \\ & & & & U_{k_2+1} & V_{k_2+1} & & \\ & & & & & \ddots & \ddots & \\ & & & & & & A_N R_{N-1}^{(p)} & B_N \\ (M_a R_1^{(1)}) & & & & & & & M_b \end{bmatrix}.$$

Column pivoting is now performed on this matrix. The columns of QAR that contain the last $(n-\ell)$ columns of U_{k_j+1}, $j = 1, \cdots, p$, are shifted to the right of the matrix, as are the columns of QAR that contain the first ℓ columns of V_{k_j-1}, $j = 2, \cdots, p+1$.

Denoting the overall pivot matrix by P_c, we write the pivoted form as

$$QARP_c = \left[\begin{array}{ccccc|cccccc}
T_1 & & & & & S_1 & & & & & \\
\hat{A}_1^{(2)} & \hat{B}_1^{(1)} & & & & \hat{A}_1^{(1)} & \hat{B}_1^{(2)} & & & & \\
 & T_2 & & & & & S_2 & & & & \\
 & \hat{A}_2^{(2)} & \hat{B}_2^{(1)} & & & & \hat{A}_2^{(1)} & \hat{B}_2^{(2)} & & & \\
 & & T_3 & & & & & S_3 & & & \\
 & & \hat{A}_3^{(2)} & \hat{B}_3^{(1)} & & & & \hat{A}_3^{(1)} & \hat{B}_3^{(2)} & & \\
 & & & \vdots & & & & & \vdots & & \\
 & & & & T_p & & & & & S_p & \\
\hline
0 & 0 & \cdots & 0 & \hat{A}_p^{(2)} & 0 & 0 & \cdots & 0 & \hat{A}_p^{(1)} & B_N \\
\hat{M}_a^{(1)} & 0 & \cdots & 0 & 0 & \hat{M}_a^{(2)} & 0 & \cdots & 0 & 0 & M_b
\end{array}\right]. \tag{3.17}$$

The component blocks of this matrix can be described by using the following notation: $N_j = (k_j - k_{j-1} - 1)n$ denotes the dimension of the square matrix T_j, $(.)_{.,1}$ denotes the first $n - \ell$ columns of an $n \times n$ matrix, and $(.)_{.,2}$ denotes the last ℓ columns of an $n \times n$ matrix. Then

$$T_j = \begin{bmatrix} (U_{k_j+1})_{.,1} & V_{k_j+1} & & & \\ & U_{k_j+2} & V_{k_j+2} & & \\ & & \ddots & \ddots & \\ & & & U_{k_{j+1}-1} & (V_{k_{j+1}-1})_{.,2} \end{bmatrix} \in \mathbb{R}^{N_j \times N_j},$$

$$S_j = \begin{bmatrix} 0 \ (U_{k_j+1})_{.,2} \\ 0 \\ \vdots \\ 0 \\ (V_{k_{j+1}-1})_{.,1} \ 0 \end{bmatrix} \in \mathbb{R}^{N_j \times n},$$

$$\hat{B}_j^{(1)} = [(B_{k_{j+1}} R_{k_{j+1}-1}^{(j+1)})_{.,1} \ 0 \cdots 0] \in \mathbb{R}^{n \times N_{j+1}}, \qquad \hat{B}_j^{(2)} = [0 \ (B_{k_{j+1}} R_{k_{j+1}-1}^{(j+1)})_{.,2}] \in \mathbb{R}^{n \times n},$$

$$\hat{A}_j^{(1)} = [(A_{k_{j+1}} R_{k_{j+1}}^{(j)})_{.,1} \ 0] \in \mathbb{R}^{n \times n}, \qquad \hat{A}_j^{(2)} = [0 \cdots 0 \ (A_{k_{j+1}} R_{k_{j+1}}^{(j)})_{.,2}] \in \mathbb{R}^{n \times N_j},$$

$$\hat{M}_a^{(1)} = [(M_a R_1^{(1)})_{.,1} \ 0 \cdots 0] \in \mathbb{R}^{n \times N_1}, \qquad \hat{M}_a^{(2)} = [0 \ (M_a R_1^{(1)})_{.,2}] \in \mathbb{R}^{n \times n}.$$

We can now perform a block LU factorization on the remaining matrix. By restating (3.6) and (3.7) as a system of linear equations, we find that

$$T_j \hat{\Phi}_j = -S_j, \qquad j = 1, \cdots, p,$$

where

$$\hat{\Phi}_j = \begin{bmatrix} (\Phi_{k_j+1}^{(j)})_{1,.} \\ \Phi_{k_j+2}^{(j)} \\ \vdots \\ \Phi_{k_{j+1}-1}^{(j)} \\ (\Phi_{k_{j+1}}^{(j)})_{2,.} \end{bmatrix}.$$

Therefore, (3.17) can be written as the following product:

$$QARP_c = \begin{bmatrix} L^{(1)} & 0 \\ 0 & I \end{bmatrix} \left[\begin{array}{cccc|ccccc} \multicolumn{4}{c|}{U_1^{(1)}} & \multicolumn{5}{c}{U_2^{(1)}} \\ \hline 0 & \cdots & 0 & \hat{A}_p^{(2)} & 0 & \cdots & 0 & \hat{A}_p^{(1)} & B_N \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 & \hat{M}_a^{(2)} & 0 & \cdots & 0 & M_b \end{array} \right],$$

where

$$L^{(1)} = \begin{bmatrix} T_1 & & & & & & \\ & I & & & & & \\ & & T_2 & & & & \\ & & & I & & & \\ & & & & \ddots & & \\ & & & & & I & \\ & & & & & & T_p \end{bmatrix}$$

(where each I in $L^{(1)}$ is $n \times n$),

$$U_1^{(1)} = \begin{bmatrix} I & & & & \\ \hat{A}_1^{(2)} & \hat{B}_1^{(1)} & & & \\ & I & & & \\ & \hat{A}_2^{(2)} & \hat{B}_2^{(1)} & & \\ & & I & & \\ & & \hat{A}_3^{(2)} & & \\ & & & \ddots & \\ & & & & I \end{bmatrix}, \quad U_2^{(1)} = \begin{bmatrix} -\hat{\Phi}_1 & & & & \\ \hat{A}_1^{(1)} & \hat{B}_1^{(2)} & & & \\ & -\hat{\Phi}_2 & & & \\ & \hat{A}_2^{(1)} & \hat{B}_2^{(2)} & & \\ & & -\hat{\Phi}_3 & & \\ & & \hat{A}_3^{(1)} & & \\ & & & \ddots & \\ & & & -\hat{\Phi}_p & 0 \end{bmatrix}.$$

Now

$$\begin{aligned} U_1^{(1)} &= \begin{bmatrix} I & & & & & & \\ \hat{A}_1^{(2)} & I & \hat{B}_1^{(1)} & & & & \\ & & I & & & & \\ & & \hat{A}_2^{(2)} & I & \hat{B}_2^{(1)} & & \\ & & & & I & & \\ & & & & \hat{A}_3^{(2)} & & \\ & & & & & \ddots & \\ & & & & & & I \end{bmatrix} \begin{bmatrix} I & & & & \\ 0 & 0 & & & \\ & I & & & \\ & 0 & 0 & & \\ & & I & & \\ & & & \ddots & \\ & & & & I \end{bmatrix} \\ &\triangleq L^{(2)} U_1^{(2)}. \end{aligned}$$

A little manipulation using the definitions of the coefficient matrices for the reduced system (3.15), the definitions of $\hat{A}_j^{(1)}$, $\hat{A}_j^{(2)}$, $\hat{B}_j^{(1)}$, and $\hat{B}_j^{(2)}$, and the fundamental boundary conditions (3.5) yields

$$-\hat{A}_j^{(2)}\hat{\Phi}_j + \tilde{A}_j = \hat{A}_j^{(1)}, \qquad -\hat{B}_j^{(1)}\hat{\Phi}_{j+1} + \tilde{B}_j = \hat{B}_j^{(2)}.$$

Hence

$$
\begin{aligned}
U_2^{(1)} &= \begin{bmatrix} I & & & & & & \\ \hat{A}_1^{(2)} & I & \hat{B}_1^{(1)} & & & & \\ & & I & & & & \\ & & \hat{A}_2^{(2)} & I & \hat{B}_2^1 & & \\ & & & & I & & \\ & & & & & \ddots & \\ & & & & & & I \end{bmatrix} \begin{bmatrix} -\hat{\Phi}_1 & & & & & \\ \tilde{A}_1 & \tilde{B}_1 & & & & \\ & -\hat{\Phi}_2 & & & & \\ & \tilde{A}_2 & \tilde{B}_2 & & & \\ & & -\hat{\Phi}_3 & & & \\ & & & \ddots & & \\ & & & & -\hat{\Phi}_p & 0 \end{bmatrix} \\
&\triangleq L^{(2)} U_2^{(2)}.
\end{aligned}
$$

Using the definitions of $U_2^{(2)}$ and $U_1^{(2)}$ and of $\hat{M}_a^{(1)}$, $\hat{M}_a^{(2)}$, $\hat{M}_b^{(1)}$, and $\hat{M}_b^{(2)}$, we have that

$$
\begin{bmatrix} 0 & \cdots & 0 & \hat{A}_p^{(2)} \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 \end{bmatrix} U_1^{(2)} = \begin{bmatrix} 0 & \cdots & 0 & \hat{A}_p^{(2)} \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 \end{bmatrix}
$$

($U_1^{(2)}$ has the effect of deleting a number of the zero columns) and

$$
\begin{aligned}
&\begin{bmatrix} 0 & \cdots & 0 & \hat{A}_p^{(2)} \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 \end{bmatrix} U_2^{(2)} + \begin{bmatrix} 0 & \cdots & 0 & \tilde{A}_p & B_N \\ \tilde{M}_a & 0 & \cdots & 0 & M_b \end{bmatrix} \\
&= \begin{bmatrix} 0 & \cdots & 0 & \hat{A}_p^{(1)} & B_N \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 & M_b \end{bmatrix}.
\end{aligned}
$$

Hence, by modifying the factorization (3.17), we find that

$$
QARP_c = \left[\begin{array}{cccc|c} \multicolumn{4}{c|}{L^{(1)}L^{(2)}} & 0 \\ \hline 0 & \cdots & 0 & \hat{A}_p^{(2)} & \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 & I \end{array}\right] \left[\begin{array}{c|ccccc} U_1^{(2)} & \multicolumn{5}{c}{U_2^{(2)}} \\ \hline & 0 & \cdots & 0 & \tilde{A}_p & B_N \\ 0 & \tilde{M}_a & 0 & \cdots & 0 & M_b \end{array}\right].
$$

Now, let P_r be a permutation matrix that pivots all zero rows of $U_1^{(2)}$ to the bottom:

$$
P_r U_1^{(2)} = P_r \begin{bmatrix} I & & & & \\ 0 & 0 & & & \\ & I & & & \\ & 0 & 0 & & \\ & & I & & \\ & & & \ddots & \\ & & & & I \end{bmatrix} = \left[\begin{array}{c} I \\ \hline 0 \end{array}\right].
$$

Then, augmenting P_r with a $2n \times 2n$ identity matrix to obtain

$$
\tilde{P}_r = \begin{bmatrix} P_r & \\ & I \end{bmatrix},
$$

we have that

$\tilde{P}_r QARP_c$

$$= \tilde{P}_r \left[\begin{array}{cccc|c} \multicolumn{4}{c|}{L^{(1)}L^{(2)}} & 0 \\ \hline 0 & \cdots & 0 & \hat{A}_p^{(2)} & \\ \hat{M}_a^{(1)} & 0 & \cdots & 0 & I \end{array}\right] \tilde{P}_r^T \tilde{P}_r \left[\begin{array}{c|ccccc} U_1^{(2)} & \multicolumn{5}{c}{U_2^{(2)}} \\ \hline & 0 & \cdots & 0 & \tilde{A}_p & B_N \\ & \tilde{M}_a & 0 & \cdots & 0 & M_b \end{array}\right]$$
$$= \begin{bmatrix} L^{(3)} & 0 \\ L^{(4)} & I \end{bmatrix} \begin{bmatrix} I & U_2^{(3)} \\ 0 & \tilde{A} \end{bmatrix},$$

where

$$L^{(3)} = \begin{bmatrix} T_1 & & & \\ & T_2 & & \\ & & \ddots & \\ & & & T_p \end{bmatrix}, \qquad L^{(4)} = \begin{bmatrix} \hat{A}_1^{(2)} & \hat{B}_1^{(1)} & & & \\ & \hat{A}_2^{(2)} & \hat{B}_2^{(1)} & & \\ & & \ddots & & \\ & & & & \hat{A}_p^{(2)} \\ \hat{M}_a^{(1)} & & & & 0 \end{bmatrix},$$

$$U_2^{(3)} = \begin{bmatrix} -\hat{\Phi}_1 & & & & \\ & -\hat{\Phi}_2 & & & \\ & & \ddots & & \\ & & & -\hat{\Phi}_p & 0 \end{bmatrix}, \qquad \tilde{A} = \begin{bmatrix} \tilde{A}_1 & \tilde{B}_1 & & & \\ & \tilde{A}_2 & \tilde{B}_2 & & \\ & & \ddots & \ddots & \\ & & & \tilde{A}_p & B_N \\ \tilde{M}_a & & & & M_b \end{bmatrix}.$$

Finally, since linear systems involving each matrix T_j (and hence $L^{(3)}$) can be solved by a triangular substitution process, it follows that there are permutation matrices P_0 and P_1 such that $P_0 L^{(3)} P_1$ is lower triangular. Hence

$$\begin{bmatrix} P_0 & \\ & I \end{bmatrix} \tilde{P}_r Q A R P_c \begin{bmatrix} P_1 & \\ & I \end{bmatrix}$$
$$(3.18) = \begin{bmatrix} P_0 & \\ & I \end{bmatrix} \begin{bmatrix} L^{(3)} & 0 \\ L^{(4)} & I \end{bmatrix} \begin{bmatrix} P_1 & \\ & I \end{bmatrix} \begin{bmatrix} P_1^T & \\ & I \end{bmatrix} \begin{bmatrix} I & U_2^{(3)} \\ 0 & \tilde{A} \end{bmatrix} \begin{bmatrix} P_1 & \\ & I \end{bmatrix}$$
$$= \begin{bmatrix} P_0 L^{(3)} P_1 & 0 \\ L^{(4)} P_1 & I \end{bmatrix} \begin{bmatrix} I & P_1^T U_2^{(3)} \\ 0 & \tilde{A} \end{bmatrix}.$$

By making the obvious identifications

$$P_L = \begin{bmatrix} P_0 & \\ & I \end{bmatrix} \tilde{P}_r, \qquad P_R = P_c \begin{bmatrix} P_1 & \\ & I \end{bmatrix},$$
$$\hat{L}_1 = P_0 L^{(3)} P_1, \qquad \hat{L}_2 = L^{(4)} P_1, \qquad \hat{U}_2 = P_1^T U_2^{(3)},$$

we obtain (3.16).

To complete the proof, we note that the quantities in both factors on the right-hand side of (3.16) are quantities that are *actually computed* by Algorithm PSC. The remainder of the algorithm consists of

1. performing the orthogonal "preprocessing" of A with Q and R, and operating with Q on the right-hand side c;
2. permuting the rows and columns of QAR and the rows of Qc in a predetermined way;
3. performing an LU factorization with no further pivoting, *stopping the elimination at stage* $(N+1-p)n$;

4. doing a "forward substitution" with $\hat{L}_2$ to get the right-hand side of the reduced system;
5. continuing the forward substitution with $\hat{L}_1$ to get the particular solution $\hat{z}_i$;
6. solving the reduced system (coefficient matrix $\tilde{A}$) to obtain the s_j; and
7. doing a back substitution, followed by orthogonal transformations, to recover all the x_i.

■

Stability of the algorithm can now be proved by using error analysis techniques from numerical linear algebra. We have the following result.

THEOREM 3.3. *Suppose that* PSC *is used to solve (1.4) in finite-precision floating-point arithmetic, with unit roundoff* $u \ll 1$. *Assume that*

(i) the growth in the fundamental solution is not excessive; that is, there is a moderate constant γ_1 *such that*

$$\max_{j=1,\cdots,p} \max_{i=k_{j-1}+1,\cdots,k_j} \|\Phi_i^{(j)}\|_2 \le \gamma_1, \qquad \max_{j=1,\cdots,p} \|T_j^{-1}\|_2 \le \gamma_1;$$

(ii) the reduced system $\tilde{A}s = \tilde{c}$ *is solved in a stable way; that is, there is a moderate constant* γ_2 *such that the computed solution* $\bar{s}$ *satisfies*

$$(\tilde{A} + E_{\tilde{A}})\bar{s} = (\tilde{c} + e_{\tilde{c}}), \tag{3.19}$$

where

$$\|E_{\tilde{A}}\|_2 \le \gamma_2 u \|\tilde{A}\|_2, \qquad \|e_{\tilde{c}}\|_2 \le \gamma_2 u \|\tilde{c}\|_2. \tag{3.20}$$

Then the computed solution $\bar{x}$ *of* $Ax = b$ *satisfies*

$$(A + E_A)\bar{x} = (b + e_b), \tag{3.21}$$

where

$$\|E_A\|_2 \le \gamma_3 u \|A\|_2, \qquad \|e_b\|_2 \le \gamma_4(1 + \|A\|_2) u \|b\|_2, \tag{3.22}$$

where $\gamma_3 = O(Nn^{5/2} + N^3 np^{-1/2}\gamma_1^2\gamma_2)$ *and* $\gamma_4 = O(n^2 p^{1/2} N \gamma_1 \gamma_2)$.

Proof. We assume throughout the proof that $\|.\|$ denotes the Euclidean norm $\|.\|_2$. Our result depends on standard analysis for Householder QR factorizations (Lawson and Hanson, [8, pp. 86–89]) and LU factorizations (Golub and Van Loan [6, §3.3]). The key to the stability argument is the fact that element growth in the L and U factors of $P_L Q A R P_R$ is bounded, because of (i). We stress that our assumption (i) is reasonable: when (1.4) is well conditioned, a dichotomy exists, and so our use of decoupling will ensure that (i) holds provided that the starting matrices $R_{k_j+1}^{(j)}$ for each partition are chosen appropriately.

First, we take account of the errors arising from the initial orthogonal transformation. By the argument of Lawson and Hanson [8, p. 87], we have that if $C \in \mathbb{R}^{n \times n}$ is a general matrix and $Q \in \mathbb{R}^{n \times n}$ is a product of n Householder transformations, then

$$\text{comp}(QC) = Q(C + H),$$

where comp(.) denotes the computed value of its argument, taking roundoff error into account, and

$$\|H\| \le (3n + 40)n^{3/2}\text{u}\|C\| + O(\text{u}^2).$$

For the case of a vector $c \in \mathbf{R}^n$,

$$\mathrm{comp}(Qc) = Q(c+h),$$

where

$$\|h\| \le (3n+40)n\mathrm{u}\|c\| + O(\mathrm{u}^2).$$

Applying these results to the initial orthogonal transformation of A, we find that

$$\begin{aligned} U_i &= \mathrm{comp}(Q_i^T \mathrm{comp}(A_i R_i)) \\ &= Q_i^T \left[(A_i + H_i^1)R_i + H_i^2\right] \\ &= Q_i^T [A_i + H_i^3] R_i, \end{aligned}$$

where

$$\|H_i^3\| \le \|H_i^1\| + \|H_i^2\| \le 2(3n+40)n^{3/2}\|A_i\|\mathrm{u} + O(\mathrm{u}^2).$$

Similarly,

$$V_i = Q_i^T [B_i + H_i^4] R_{i+1},$$

where

$$\|H_i^4\| \le 2(3n+40)n^{3/2}\|B_i\|\mathrm{u} + O(\mathrm{u}^2).$$

Similar analysis can be applied to $\mathrm{comp}(M_a R_1^{(1)})$ and to the block rows k_j that are not multiplied from the left by an orthogonal transformation. Since the pivoting operations do not incur any roundoff error,

$$\mathrm{comp}(P_L Q A R P_R) = P_L Q(A + \tilde{H}) R P_R, \tag{3.23}$$

where

$$\begin{aligned} \|\tilde{H}\| &\le (N+1)\left[\sup_i \|H_i^3\| + \sup_i \|H_i^4\|\right] \\ &\le 4(N+1)n^{3/2}(3n+40)\|A\|\mathrm{u} + O(\mathrm{u}^2). \end{aligned}$$

Similarly,

$$\mathrm{comp}(Qb) = Q(b+h),$$

where

$$\|h\| \le (N+1)^{1/2}(3n+40)n\|b\|\mathrm{u} + O(\mathrm{u}^2).$$

We now examine the LU factorization of $\mathrm{comp}(P_L Q A R P_R)$. Although the Gaussian elimination process is terminated prematurely, we can use the proof of Theorem 3.3.1 of Golub and Van Loan [6] to show that

$$\tilde{L}\tilde{U} = \mathrm{comp}(P_L Q A R P_R) + \hat{H}, \tag{3.24}$$

where

$$|\hat{H}| \le 3(N+1)n\mathrm{u}\left(|\mathrm{comp}(P_L Q A R P_R)| + |\tilde{L}||\tilde{U}|\right) + O(\mathrm{u}^2).$$

Hence,

$$\|\hat{H}\|_2 \le 3(N+1)n\mathrm{u}\left[\|A\| + \|\tilde{L}\|\|\tilde{U}\|\right] + O(\mathrm{u}^2). \tag{3.25}$$

Now

$$\|\tilde{L}\| \le \|\hat{L}_1\| + \|\hat{L}_2\| + \|\tilde{A}\| = \|L^{(3)}\| + \|L^{(4)}\| + \|\tilde{A}\|, \tag{3.26}$$

and so, by using the definitions that appear during the proof of Theorem 3.2,

$$\begin{aligned} \|L^{(3)}\| &\le \|\mathrm{comp}(QAR)\| \le \|A\| + O(\mathrm{u}), \\ \|L^{(4)}\| &\le 2(p+1)^{1/2}\max\left(\max_i(\|A_iR_i\|, \|B_iR_{i+1}\|), \|M_aR_1\|\right) + O(\mathrm{u}) \\ &\le 2(p+1)^{1/2}\|A\| + O(\mathrm{u}), \\ \|\tilde{A}\| &\le 2(p+1)^{1/2}\max\left(\max_j(\|\tilde{A}_j\|, \|\tilde{B}_j\|), \|\tilde{M}_a\|, \|M_b\|\right). \end{aligned}$$

In the last of these inequalities,

$$\|\tilde{A}_j\| \le \|A_{k_j+1}\|(1 + \|\hat{\Phi}_j\| + O(\mathrm{u})) \le \|A\|(1 + (N/p)\gamma_1 + O(\mathrm{u})).$$

Similar inequalities can be derived for $\|\tilde{B}_j\|$ and $\|\tilde{M}_a\|$, so

$$\|\tilde{A}\| \le 2(p+1)^{1/2}(1 + (N/p)\gamma_1)\|A\| + O(\mathrm{u}).$$

For the upper triangular factor, we have that

$$\begin{aligned} \|\tilde{U}\| &\le 2 + \|\hat{U}_2\| = 2 + \|U_2^{(3)}\| \\ &\le 2 + \max_j \|\hat{\Phi}_j\| + O(\mathrm{u}) \le 2 + (N/p)\gamma_1 + O(\mathrm{u}). \end{aligned} \tag{3.27}$$

By using some elementary inequalities, we then have from (3.25), (3.26), and (3.27) that

$$\|\hat{H}\| \le 6(N+1)(p+1)^{1/2}n(3 + (N/p)\gamma_1)^2\|A\|\mathrm{u} + O(\mathrm{u}).$$

Consider now the forward and back substitution process involving $\tilde{L}$ and $\tilde{U}$, that is,

$$\tilde{L}\tilde{U}y = \mathrm{comp}(P_LQb) = P_LQ(b + \tilde{h}). \tag{3.28}$$

Using (3.19), (3.20), and a simple backwards error argument, we can show that the computed solution $\bar{y}$ of (3.28) will satisfy

$$\begin{aligned} &\begin{bmatrix} \hat{L}_1 + \hat{H}_1 & 0 \\ \hat{L}_2 + \hat{H}_2 & \tilde{A} + E_{\tilde{A}} \end{bmatrix} \begin{bmatrix} I & \hat{U}_2 + \hat{H}_3 \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \end{bmatrix} = \begin{bmatrix} \bar{Q}_1(b + \tilde{h}) \\ \bar{Q}_2(b + \tilde{h}) + \tilde{e} \end{bmatrix} \\ \Longrightarrow \quad &\left(\tilde{L} + E_{\tilde{L}}\right)\left(\tilde{U} + E_{\tilde{U}}\right)\bar{y} = P_LQ(b + \tilde{h} + e_{\tilde{b}}), \end{aligned} \tag{3.29}$$

where $\bar{Q}_1$ consists of the first $(N-p)n$ rows of P_LQ, $\bar{Q}_2$ contains the last $(p+1)n$ rows, and $E_{\tilde{L}}$, $E_{\tilde{U}}$, and $e_{\tilde{b}}$ are defined in obvious ways. Now (3.19) and (3.20) imply that

$$\begin{aligned} \|\tilde{e}\| &\le \gamma_2\mathrm{u}\left\|Q_1(b + \tilde{h}) - (\hat{L}_2 + \hat{H}_2)(\hat{L}_1 + \hat{H}_1)^{-1}Q_2(b + \tilde{h})\right\| \\ &\le \gamma_2\mathrm{u}\left(1 + \|\hat{L}_2\|\|\hat{L}_1^{-1}\| + O(\mathrm{u})\right)(\|b\| + O(\mathrm{u})) \\ &\le \bar{\gamma}_4\mathrm{u}\|A\|\|b\| + O(\mathrm{u}^2), \end{aligned}$$

where $\bar\gamma_4 = O(\gamma_1\gamma_2 p^{1/2})$, and

$$\|E_{\tilde A}\| \le \gamma_2 \mathrm{u}\|\tilde A\| \le 2\gamma_2(p+1)^{1/2}(1+(N/p)\gamma_1)\mathrm{u}\|A\| + O(\mathrm{u}^2).$$

Following Golub and Van Loan [6, p. 106], we find that the remaining error terms in (3.29) satisfy

$$\begin{aligned}
\|\hat H_1\| &\le n\mathrm{u}\|\hat L_1\| \le n\mathrm{u}\|A\| + O(\mathrm{u}^2),\\
\|\hat H_2\| &\le n\mathrm{u}\|\hat L_2\| \le 2n\mathrm{u}(p+1)^{1/2}\|A\| + O(\mathrm{u}^2),\\
\|\hat H_3\| &\le n\mathrm{u}\|\hat U_2\| \le 2n\mathrm{u}(N/p)\gamma_1.
\end{aligned}$$

Hence

$$\begin{aligned}
\|E_{\tilde L}\| &\le \|\hat H_1\| + \|\hat H_2\| + \|E_{\tilde A}\| \le \bar\gamma_3\mathrm{u}\|A\| + O(\mathrm{u}^2),\\
\|E_{\tilde U}\| &\le 2nN\gamma_1\mathrm{u},\\
\|e_{\tilde b}\| &\le \bar\gamma_4\mathrm{u}\|A\|\|b\|,
\end{aligned}$$

where $\bar\gamma_3 = O(np^{-1/2}N\gamma_1\gamma_2)$. From (3.23), (3.24), and (3.29), we have that

$$\begin{aligned}
\left(\tilde L\tilde U + E_{\tilde L}\tilde U + \tilde L E_{\tilde U} + O(\mathrm{u}^2)\right)\bar y &= P_L Q(b+\tilde h + e_{\tilde b})\\
\Longrightarrow (P_L QARP_R + \bar H)\,\bar y &= P_L Q(b+\tilde h + e_{\tilde b}),
\end{aligned}$$

where

$$\bar H = P_L Q\tilde H RP_R + \hat H + E_{\tilde L}\tilde U + \tilde L E_{\tilde U} + O(\mathrm{u}^2).$$

Hence

$$\|\bar H\| \le \|\tilde H\| + \|\hat H\| + \|E_{\tilde L}\|\|\tilde U\| + \|\tilde L\|\|E_{\tilde U}\| \le \gamma_3,$$

where γ_3 is as defined in the statement of the theorem. Also,

$$\|\tilde h + e_{\tilde b}\| \le \gamma_4\|b\|.$$

The result follows by making the identifications

$$E_A = Q^T P_L^T \bar H P_R^T R^T, \qquad e_b = \tilde h + e_{\tilde b}, \qquad \bar y = RP_R\bar y.$$

■

If the algorithm of this section is applied recursively (that is, if the reduced system is itself solved on multiple processors by using the same technique), the theorem indicates the amount of deterioration in accuracy that can be expected in moving between levels of the recursion.

4. A No-fill-in Variant. In the preceding section, we showed how the fundamental and particular solutions for the problem (1.4) could be calculated by using the blocks U_i and V_i from the transformed coefficient matrix. As described there, the algorithm assumed that the data for the initial orthogonal reduction (that is, the U_i, V_i, Q_i, and R_i matrices) is not overwritten in storage. This assumption means that new storage must be used to store the fundamental solution blocks Φ_i, resulting in a fill-in of approximately 25%.

In this section, we outline a scheme in which the fill-in is reduced by storing the components of Φ_i in storage formerly occupied by U_i and V_i. At the same time, we wish to allow for the possibility that the particular solution is calculated at some later time than the fundamental solution, that is, the right-hand sides c_i, $i = 1, \cdots, N$ and d are not known at the time the fundamental solution is calculated. This situation may arise when a "chord method" approach is applied to a nonlinear version of (1.4). In this method, approximate Newton iterations are calculated by using the same Jacobian information for a number of successive iterations, while the right-hand side changes from iteration to iteration. In the preceding section, particular solutions are calculated by using (3.9) and (3.10). Hence, if we plan to overwrite components of U_i and V_i, we need to devise new formulae for finding the $\hat{z}_i$s.

To satisfy both requirements of the preceding paragraph, we need to define a "general solution" $\Psi_i^{(j)}$, $i = k_j + 1, \cdots, k_{j+1}$ on partition j in such a way that each $\Psi_i^{(j)} \in \mathbb{R}^{n\times n}$ is nonsingular. Moreover, $\Psi_i^{(j)}$ will be identical to the fundamental solution $\Phi_i^{(j)}$ whenever all U_i and V_i in partition j are nonsingular. Computation of the sequence Ψ_i involves a simple modification of the forward sweep/reverse sweep process (3.6) and (3.7). The only enhancement is that when some $(U_i)_{22}$ or $(V_i)_{11}$ are singular, diagonal terms are added to $(U_i)_{22}(\Psi_i)_{22}$ or $(V_i)_{11}(\Psi_{i+1})_{11}$ to prevent singularity in the next general solution matrix in the sequence. Note that we are taking advantage here of the fact that U_i and V_i (and also Φ_i and Ψ_i) are upper triangular. This means that singularity manifests itself as zero elements on the diagonal and is easily remedied by replacing these zeros by, say, ones, which is what we do.

For $i = k_j + 1, \cdots, k_{j+1} - 1$, we define

$$Z_i^F = [e_l^\ell]_{l\in\mathcal{A}_i^F},$$

where e_l^ℓ is the $\mathbb{R}^\ell$ unit vector, with zeros everywhere except for a 1 in position l, and

$$\mathcal{A}_i^F = \{l \mid l\text{-th diagonal element of } (U_i)_{22} \text{ is zero}\}.$$

Note that $Z_i^F(Z_i^F)^T$ is an $\ell \times \ell$ matrix that is zero everywhere except for ones in the diagonal positions in which $(U_i)_{22}$ has zeros. Similarly, define

$$Z_i^R = [e_l^{n-\ell}]_{l\in\mathcal{A}_i^R},$$

where $e_l^{n-\ell}$ is the $\mathbb{R}^{n-\ell}$ unit vector with a 1 in the l position, and

$$\mathcal{A}_i^R = \{l \mid l\text{-th diagonal element of } (V_i)_{11} \text{ is zero}\}.$$

The boundary conditions for Ψ_i coincide with (3.5); that is,

$$(4.1) \qquad (\Psi_{k_j+1})_{2,\cdot} = [0 \mid I], \qquad (\Psi_{k_{j+1}})_{1,\cdot} = [I \mid 0].$$

The sweeps are defined as follows:

for $i = k_j + 1, \cdots, k_{j+1} - 1$

$$(4.2) \qquad (\Psi_{i+1})_{2,\cdot} = -(V_i)_{22}^{-1}\left\{[0 \mid (U_i)_{22}(\Psi_i)_{22}] + Z_i^F \begin{bmatrix} 0 \\ Z_i^F \end{bmatrix}^T\right\};$$

for $i = k_{j+1} - 1, \cdots, k_j + 1$

$$(\Psi_i)_{1,\cdot} =$$

$$(4.3)\quad -(U_i)_{11}^{-1}\left\{(V_i)_{11}(\Psi_{i+1})_{1,\cdot} + (V_i)_{12}(\Psi_{i+1})_{2,\cdot} + (U_i)_{12}(\Psi_i)_{2,\cdot} + Z_i^R\begin{bmatrix} Z_i^R \\ 0 \end{bmatrix}^T\right\}.$$

Note that the extra terms in (4.2) and (4.3) do not disturb the upper triangularity of each Ψ_i.

By making use of the relationship between the fundamental and general solution, we can store the general solution in a compressed format. We show how this is done for the $(2,2)$ block of Ψ_i; for the other two nonzero blocks the technique is similar.

First, we show that we can write

$$(4.4)\qquad (\Psi_i)_{22} = (\Phi_i)_{22} + \bar{W}_i^F(\bar{Z}_i^F)^T,$$

where $\bar{Z}_i^F$ contains all e_l^ℓ such that the l-th diagonal of $(U_k)_{22}$ is zero for some, and possibly more than one, $k = k_j + 1, \cdots, i-1$. For $i = k_j + 1$, $\bar{W}_i^F$ and $\bar{Z}_i^F$ are null. Assuming that (4.4) is true for $k = k_j + 1, \ldots, i$, we have from (4.2) that

$$\begin{aligned}(\Psi_{i+1})_{22} &= -(V_i)_{22}^{-1}\left\{(U_i)_{22}\left[(\Phi_i)_{22} + \bar{W}_i^F(\bar{Z}_i^F)^T\right] + Z_i^F\begin{bmatrix} 0 \\ Z_i^F \end{bmatrix}^T\right\} \\ &= (\Phi_{i+1})_{22} - (V_i)_{22}^{-1}(U_i)_{22}\bar{W}_i^F(\bar{Z}_i^F)^T - (V_i)_{22}^{-1}Z_i^F(Z_i^F)^T.\end{aligned}$$

We obtain $\bar{Z}_{i+1}^F$ by merging the columns of $\bar{Z}_i^F$ and Z_i^F, while $\bar{W}_{i+1}^F$ is obtained by merging $-(V_i)_{22}^{-1}(U_i)_{22}\bar{W}_i^F$ and $-(V_i)_{22}^{-1}Z_i^F$. We "merge" rather than simply append Z_i^F to $\bar{Z}_i^F$ since there is no need for $\bar{Z}_{i+1}^F$ to have two copies of the same column. (The same effect can be obtained by adding the two corresponding columns of $\bar{W}_{i+1}^F$.) The corresponding formula to (4.4) for the $(1,1)$ and $(1,2)$ blocks is

$$(4.5)\qquad (\Psi_i)_{1,\cdot} = (\Phi_i)_{1,\cdot} + \bar{W}_i^R(\bar{Z}_i^R)^T.$$

In each case, the substantive additional storage requirements are for the matrices $\bar{W}_i^F$ and $\bar{W}_i^R$ (the $\bar{Z}_i$ matrices can be stored in a few integer locations.) In the worst case, this will require the same amount of storage as the general solution itself, but we usually expect it to be much less. For example, if the (ℓ,ℓ) element of one or more of the $(U_i)_{22}$ matrices is zero, then we need about $(k_{j+1} - k_j - 1)n$ locations to store the $\bar{W}_i$, or about $2/n$ of the space required by the entire general solution.

As we noted earlier, a particular solution of the recurrence (3.3) can be calculated by performing the forward and backward sweeps (3.9) and (3.10). Since we would like to overwrite some components of the U_i and V_i matrices by components of Φ_i, we now describe an alternative method for calculating the $\hat{z}_i$ which makes use of the Ψ_i but only of selected components of U_i and V_i.

The boundary conditions (3.8) are used for $\hat{z}_i$, $i = k_j + 1, \cdots, k_{j+1}$, as before. Defining the change of variables

$$(4.6)\qquad \hat{z}_i = \Psi_i v_i,$$

and using the boundary conditions (4.1), we obtain

$$(4.7)\qquad (v_{k_j+1})_{2,\cdot} = 0, \qquad (v_{k_{j+1}})_{1,\cdot} = 0.$$

We now substitute in (3.3) and define $\hat{c}_i = Q_i^T c_i$ to obtain

$$U_i \Psi_i v_i + V_i \Psi_{i+1} v_{i+1} = \hat{c}_i. \tag{4.8}$$

Note from (4.2) and (4.3) that

$$-U_i \Psi_i = V_i \Psi_{i+1} + \begin{bmatrix} Z_i^R (Z_i^R)^T & 0 \\ 0 & Z_i^F (Z_i^F)^T \end{bmatrix}. \tag{4.9}$$

By isolating the last ℓ rows of (4.8), and by considering the (2,2) block of (4.9), we obtain

$$\begin{aligned} &-[(V_i)_{22}(\Psi_{i+1})_{22} + Z_i^F (Z_i^F)^T](v_i)_2 + (V_i)_{22}(\Psi_{i+1})_{22}(v_{i+1})_2 = (\hat{c}_i)_2 \\ &\Rightarrow \quad (v_{i+1})_2 = (v_i)_2 + [(V_i)_{22}(\Psi_{i+1})_{22}]^{-1}[(\hat{c}_i)_2 + Z_i^F (Z_i^F)^T (v_i)_2]. \end{aligned} \tag{4.10}$$

Here, $(.)_2$ denotes the last ℓ rows of a vector in $\mathbb{R}^n$. By isolating the first $n-\ell$ rows of the expressions (4.8) and (4.9), we find that

$$\begin{aligned} (v_i)_1 &= (v_{i+1})_1 + [(U_i)_{11}(\Psi_i)_{11}]^{-1} \{ Z_i^R (Z_i^R)^T (v_{i+1})_1 + (\hat{c}_i)_1 \\ &\quad + [(U_i)_{11}(\Psi_i)_{12} + (U_i)_{12}(\Psi_i)_{22}][(v_{i+1})_2 - (v_i)_2] \}. \end{aligned} \tag{4.11}$$

We conclude that v_i, $i = k_j + 1, \cdots, k_{j+1}$, can be found by doing a forward sweep using (4.10) followed by a reverse sweep using (4.11). Note that we need to solve linear systems with coefficient matrices $(V_i)_{22}$, $(U_i)_{11}$, $(\Psi_i)_{11}$, and $(\Psi_i)_{22}$ in order to obtain $\hat{z}_i$ from (4.6) and (4.10),(4.11). If we assume that (1.4) is well conditioned and that the partitioning is done correctly, these matrices are invertible.

The blocks $(U_l)_{22}$, $(V_l)_{11}$, and $(V_l)_{12}$ are not used in (4.10) and (4.11). We can therefore overwrite these blocks in storage to avoid fill-in. During the forward sweep (3.6), $(U_i)_{22}$ can be progressively overwritten by $(V_i)_{22}$ in memory, and $(\Phi_{i+1})_{22}$ can be stored in the space vacated by $(V_i)_{22}$. During the reverse sweep (3.7), $(\Phi_i)_{11}$ and $(\Phi_i)_{12}$ can progressively overwrite $(V_i)_{11}$ and $(V_i)_{12}$. The matrices $\bar{W}_i^F$ and $\bar{W}_i^R$ which are needed to recover Ψ_i from Φ_i will need to be stored in new locations.

5. Extension to Problems with Multipoint Conditions or Parameters.

We now show how the algorithm of §3 can be modified to handle problems of the forms (1.5) and (1.6). As mentioned in §2, the fundamental solution modes in these problems exhibits not dichotomy but, in general, polychotomy and skew-polychotomy. The practical consequence for the Algorithm PSC is that the value of ℓ (the number of "decreasing" fundamental modes) is no longer constant across all partitions; in fact, it may increase or decrease repeatedly within each partition. It is even possible for the submatrix $[A_i \; B_i]$ to be rank deficient. In order to adapt PSC to these circumstances, we need to be able to recognize when ℓ has changed, and to modify the compactification strategy accordingly.

A change in ℓ is recognized by periodically examining the diagonal elements of $V_i^{-1}U_i$ during the initial orthogonal factorization process (3.2). As mentioned earlier, these are $(U_i)_{ll}/(V_i)_{ll}$, $l = 1, \ldots, n$. If the number of diagonals that are less than 1 is either greater than or less than the current value of ℓ for a few successive stages, then we deem ℓ to have changed. The most general way of handling such a dichotomy change is simply to break off the present partition at the current stage point, and start a new one. Specifically, suppose that at stage i during the orthogonal preprocessing of partition j, we decide that ℓ has changed. We then set $k_{j+} = i + 1$ and terminate the preprocessing of the current partition after calculating $R^{(j)}_{k_{j+}}$. Next, we skip a row and start a new partition by choosing $R^{(j+)}_{k_{j+}+1}$ as described in §3. The skipped row must now be added to the reduced system. If $s_{j+} \in \mathbb{R}^n$ is the "reduced" variable for the new partition, the extra equation is

$$A_{k_{j+}} R^{(j)}_{k_{j+}} \left[\Phi^{(j)}_{k_{j+}} s_j + \hat{z}^{(j)}_{k_{j+}}\right] + B_{k_{j+}} R^{(j+)}_{k_{j+}+1} \left[\Phi^{(j+)}_{k_{j+}+1} s_{j+} + \hat{z}^{(j+)}_{k_{j+}+1}\right] = c_{k_{j+}}.$$

To avoid creating a new partition in the case in which ℓ increases, we can pick up an extra component during the forward sweep calculation of the "decreasing" part of Φ_i, at the point at which the dichotomy change occurs. Correspondingly, a row is dropped from the first part of Φ_i during the backward sweep. This is essentially the strategy used by de Hoog and Mattheij [4, §4].

A circumstance that causes more immediate failure of the algorithm occurs when the submatrix $[A_i \; B_i]$ fails to have full rank for some i. When this occurs, $[U_i \; V_i]$ is also rank deficient, and it is easy to show that for at least one index $l = 1, \cdots, n$, the l-th diagonals of U_i and V_i are both zero. Hence at least one of $(U_i)_{11}$ and $(V_i)_{22}$ are singular, so either the forward sweep (3.6) or the reverse sweep (3.7) will break down at stage i. Since this difficulty can be detected during the initial orthogonal factorization, the fix is the same as for a dichotomy change — we create a partition break at stage i.

We assume from this point on that the number of partitions p and the separator indices k_j, $j = 1, \cdots, p$, have been altered where necessary during the partitioned compactification, to reflect the number of new dichotomy switches that were encountered.

Construction of the reduced system is slightly different for problems of the form (1.5) than it is for problems of the form (1.4). Rather than (3.11), each z_i is now expressible as

$$z_i = \Phi^{(j)}_i s_j + \Lambda^{(j)}_i \lambda + \hat{z}^{(j)}_i, \qquad x_i = R^{(j)}_i z_i, \qquad i = k_j + 1, \cdots, k_{j+1}. \tag{5.1}$$

Since

$$\begin{aligned} U_i \Phi^{(j)}_i + V_i \Phi^{(j)}_{i+1} &= 0, & i &= k_j + 1, \cdots, k_{j+1}, \\ U_i \hat{z}^{(j)}_i + V_i \hat{z}^{(j)}_{i+1} &= Q_i^T c_i, & i &= k_j + 1, \cdots, k_{j+1}, \\ U_i z_i + V_i z_{i+1} + Q_i^T C_i \lambda &= Q_i^T c_i, & i &= k_j + 1, \cdots, k_{j+1}, \end{aligned}$$

we have by substitution in (5.1) that

$$\left[U_i \Lambda^{(j)}_i + V_i \Lambda^{(j)}_{i+1} + Q_i^T C_i\right] \lambda = 0, \qquad i = k_j + 1, \cdots, k_{j+1}.$$

We choose $\Lambda^{(j)}_i$, $i = k_j + 1, \cdots, k_{j+1}$, to satisfy the recurrence suggested by this formula, namely,

$$U_i \Lambda^{(j)}_i + V_i \Lambda^{(j)}_{i+1} = -Q_i^T C_i, \qquad i = k_j + 1, \cdots, k_{j+1}. \tag{5.2}$$

Clearly, this recurrence has the same form as the one that is solved for the particular solution $\hat{z}^{(j)}_i$ (except that it has m columns instead of one), and we can solve it in exactly the same way. The choice of boundary conditions is also the same as for $\hat{z}^{(j)}_i$. When no dichotomy switching is encountered within a partition, we set

$$(\Lambda^{(j)}_{k_j+1})_{n-\ell+1:n,\cdot} = 0, \qquad (\Lambda^{(j)}_{k_{j+1}})_{1:n-\ell,\cdot} = 0. \tag{5.3}$$

In deriving the scheme (5.2) and (5.3), we are essentially treating the parameter term as a forcing term.

We can now use (5.1) to construct the reduced system. For the side conditions, we have

$$
\begin{aligned}
\sum_{i=1}^{N+1} M_i x_i + N\lambda &= d \\
\Rightarrow \sum_{j=1}^{p} \sum_{i=k_j+1}^{k_{j+1}} M_i R_i^{(j)} \left[\Phi_i^{(j)} s_j + \Lambda_i^{(j)}\lambda + \hat{z}_i^{(j)}\right] + N\lambda &= d \qquad (5.4)\\
\Rightarrow \sum_{j=1}^{p} \tilde{M}_j s_j + \tilde{N}\lambda &= \tilde{d},
\end{aligned}
$$

where

$$\tilde{M}_j = \textstyle\sum_{i=k_j+1}^{k_{j+1}} M_i R_i^{(j)} \Phi_i^{(j)}, \qquad \tilde{N} = N + \sum_{j=1}^{p} \sum_{i=k_j+1}^{k_{j+1}} M_i R_i^{(j)} \Lambda_i^{(j)},$$
$$\tilde{d} = d - \textstyle\sum_{j=1}^{p} \sum_{i=k_j+1}^{k_{j+1}} M_i R_i^{(j)} \hat{z}_i^{(j)}.$$

For the remaining equations, we obtain

$$\tilde{A}_j s_j + \tilde{B}_j s_{j+1} + \tilde{C}_j \lambda = \tilde{c}_j,$$

where $\tilde{A}_j$, $\tilde{B}_j$ and $\tilde{c}_j$ are as defined in (3.15a), and

$$\tilde{C}_j = C_{k_{j+1}} + A_{k_{j+1}} R_{k_{j+1}}^{(j)} \Lambda_{k_{j+1}}^{(j)} + B_{k_{j+1}} R_{k_{j+1}+1}^{(j+1)} \Lambda_{k_{j+1}+1}^{(j+1)}.$$

Again, recursive application of the compactification to the reduced system is possible, but only up to a point. We would expect to have an intrinsic lower bound on the number of stages in the smallest possible reduced system, namely, the total number of dichotomy switches. In other words, the smallest reduced system is one in which a dichotomy switch occurs at each stage.

For problems of the form (1.6), substitution like that described in (5.4) can be performed in the least squares objective function. We obtain

$$
\begin{aligned}
&\textstyle\sum_{i=1}^{N+1} \|M_i x_i + N_i\lambda - d_i\|^2 + \|N_0\lambda - d_0\|^2 \\
&= \textstyle\sum_{j=1}^{p} \|\hat{M}_j s_j + \hat{N}_j\lambda - \hat{d}_j\|^2 + \|N_0\lambda - d_0\|^2, \qquad (5.5)
\end{aligned}
$$

where

$$
\begin{aligned}
\hat{M}_j &= \left[M_i R_i^{(j)} \Phi_i^{(j)}\right]_{k_{j+1}}^{i=k_j+1}, \\
\hat{N}_j &= \left[N_i + M_i R_i^{(j)} \Lambda_i^{(j)}\right]_{k_{j+1}}^{i=k_j+1}, \\
\hat{d}_j &= \left[d_i - M_i R_i^{(j)} \hat{z}_i^{(j)}\right]_{k_{j+1}}^{i=k_j+1}.
\end{aligned}
$$

6. Numerical Results. We implemented the PSC and SQR algorithms on the Intel Touchstone at the California Institute of Technology. This machine has a 528-node message-passing architecture based on the Intel i860 chip, with a two-dimensional mesh interconnection configuration. Underlying the mesh is a high-speed bus which allows data to be ported between any two nodes in a fashion that is transparent to the user. Provided the total amount of data being transferred is not too great, the time to send a message across this bus is not strongly dependent on the physical locations of the source and destination nodes in the mesh. We chose this architecture since it seems typical of the new generation of massively parallel machines. The Intel Paragon will be a commercial version of the Touchstone, while the Thinking Machines CM-5 (already in production) has an essentially equivalent configuration.

Our codes handle two classes of problem — two-point boundary value problems (1.1) and (1.4), and two-point problems with parameters ((1.2) and (1.5) with $\tau_1 = a$ and $\tau_2 = b$). As discussed earlier, the SQR algorithm is similar to the one described in [12], except that Givens rotations are used in addition to Householder transformations during the QR factorizations. The extra columns in the shooting matrix that are due to the presence of parameters are handled as described in Wright [11, §5]. Each processor compresses its "slice" of the shooting matrix into a single block row, leaving a reduced system with $(p+1)n+m$ rows and columns. This system is solved by using a "cyclic reduction" variant of SQR, which we now briefly describe. Assume that the processors are numbered $0, 1, 2, \cdots, p-1$, where $p = 2^d$ for some integer $d \geq 0$. At the first level of cyclic reduction, the odd-numbered processors $2i+1$, $i = 0, 1, \cdots, p/2-1$ pass their piece of the reduced system (consisting of a single block row) to the neighboring even-numbered processor $2i$. Each processor $2i$ then compresses this block row with its own block row to produce a single block row. At the end of this first level, the size of the reduced system has been approximately halved — it now has dimension $(p/2+1)n+m$. At the second level, the processors $4i+2$ pass their data to processors $4i$. After d levels, processor 0 is left with a reduced system of dimension $2n+m$. It solves this system to produce x_1, x_{N+1}, and λ. We then backtrack along the binary tree traversed during the process of compression to recover all the intermediate solution components.

The implementation of PSC is similar. Each processor is assigned an equal-sized slice of the original shooting matrix to which it applies the stabilized compactification algorithm described in §§3 and 5. At the end of this stage, each processor contains one or, in the case of dichotomy changes, more than one block row of the reduced system. If a processor contains more than one block row, orthogonal compression like that used by SQR is applied to obtain a single block row. The cyclic reduction SQR compression scheme just described is now applied to solve the reduced system that remains.

In our tests, we form discrete problems by applying the "box scheme" to our continuous problems. We seek $x_i \in \mathbb{R}^n$, $i = 1, \cdots, N+1$ such that $x_i \approx x(t_i)$, where $t_i = a + (i-1)h$ and $h = (b-a)/N$. The parametrized ODE (1.2) is approximated by

$$(x_{i+1} - x_i)/h = A(t_{i+1/2})(x_i + x_{i+1})/2 + C(t_{i+1/2})\lambda + c(t_{i+1/2}).$$

Our first test problem has a change of dichotomy near the point $t = 1/3$.

Example 1 $n = 3$, $m = 1$.

$$(6.1)\quad \dot{x} = Q(t)\begin{bmatrix} 20 & & \\ & 10(t-1/3) & \\ & & -20 \end{bmatrix} R(t)x + C(t)\lambda + c(t), \qquad t \in [0,1],$$

Example 1, four processors, $N = 4000$. Restart points for stabilized compactification (excluding the breakpoints due to partitioning into four subintervals) and number of decreasing modes on the subinterval between this breakpoint and the preceding one.

t	.378	.403	.433	.961
No. of decreasing modes	2	1	1	1

where, using the shorthand $c = \cos t$, $s = \sin t$,

$$Q(t) = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ c-s & c+s & 1 \end{bmatrix}, \quad R(t) = \begin{bmatrix} 1 & & 1 \\ & 1 & 1 \\ & & 1 \end{bmatrix}, \quad C(t) = \begin{bmatrix} 3 \\ 0 \\ 5t \end{bmatrix}.$$

The vector function $c(t)$ is chosen so that $x(t) = e^t(1,1,1)^T$ and $\lambda = 1$ is a solution of the parametrized ODE. The boundary conditions are

$$\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \\ 2 & 3 & 4 \end{bmatrix} x(0) + \begin{bmatrix} & & 1 \\ & 1 & \\ 1 & & \\ -2 & -3 & -4 \end{bmatrix} x(1) + \begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{bmatrix} \lambda = \begin{bmatrix} 2+e \\ 1+e \\ e \\ 9-9e \end{bmatrix}.$$

PSC is able to detect the change in dichotomy near $t = 1/3$. On a single processor with $N = 1000$, a change from two decreasing modes to one decreasing mode is reported at $t = .406$. The "rightward shift" is due to the fact that our heuristic for detecting dichotomy changes is rather conservative; it reports a switch only when the behavior is consistently different over a significant number of consecutive intervals. When we use four processors, each with 1000 intervals (a total of $N = 4000$), the situation is a little more complicated. The algorithm for calculating the fundamental solution detects a number of points at which a dichotomy switch appears to occur and reports them as such. However, in all but one case, a restart at that point (as described in §5) indicates that the number of increasing and decreasing modes has not changed. The exception is, of course, the point at which the one true dichotomy switch occurs. The spurious switches do not affect the stability of the algorithm and have only a marginal effect on the computation time, since they add just a single row to the reduced system. The locations of the "breaks" are summarized in Table 6.1.

Tables 6.2 and 6.3 show timings for SQR and PSC, respectively. In both cases, the time for initial reduction of each partition remains essentially constant and is somewhat less for PSC than for SQR. The time to solve the reduced system by the cyclic reduction algorithm tends to increase as the depth of the tree increases, though not in a smooth way. Nevertheless, solution of the reduced system is such a small part of the overall computation that near-perfect speedup is attained.

Our second example arises from transport theory. Consider the single-group one-dimensional transport equation as defined by Jin and Levermore [7, Example 2]:

$$(6.2)\quad \mu\partial_x\Phi(x,\mu) + \sigma_T(x)\Phi(x,\mu) = \tfrac{1}{2}w(x)\sigma_T(x)\int_{-1}^{1}\Phi(x,\mu')\,d\mu', \qquad x \in (0,L),$$

with boundary conditions

$$(6.3)\qquad \Phi(0,\mu) = F_1(\mu)\ \ (\mu > 0), \qquad \Phi(L,\mu) = F_2(\mu)\ \ (\mu < 0).$$

Example 1. PSC: *Timings with number of intervals per processor fixed at* 1000. *All times are in seconds.*

Number of processors	1	2	4	8	16	32	64	128
Time for initial reduction	.347	.348	.347	.347	.347	.347	.347	.347
Time for reduced system solution	.001	.001	.001	.014	.013	.012	.013	.014
Total time	.371	.372	.371	.384	.382	.382	.383	.384

Example 1. SQR: *Timings with number of intervals per processor fixed at* 1000. *All times are in seconds.*

Number of processors	1	2	4	8	16	32	64	128
Time for initial reduction	.613	.613	.614	.614	.614	.613	.614	.614
Time for reduced system solution	.001	.002	.002	.012	.013	.014	.014	.014
Total time	.642	.643	.644	.654	.655	.655	.656	.656

One widely accepted method for solving this equation, the *discrete ordinates method*, proceeds by replacing the integral term in (6.2) by a quadrature approximation. The resulting two-point boundary value problem in the remaining independent variable x can then be solved to obtain an approximation to Φ. We use Gaussian quadrature with an even number of abscissae, in which specification of weights ω_i, $i = 1, \cdots, n$, and abscissae μ_i, $i = 1, \cdots, n$ with $-1 < \mu_1 < \cdots < \mu_n < 1$, $\mu_i = -\mu_{n-i+1}$, $\omega_i = \omega_{n-i+1}$, and $\omega_i > 0$ leads to the approximation

$$\int_{-1}^{1} f(\mu)\, d\mu \approx \sum_{i=1}^{n} \omega_i f(\mu_i).$$

Using the notation

$$\Phi_i(x) \triangleq \Phi(x, \mu_i),$$

and discretizing the boundary conditions in an obvious (though not optimal) way, we can write the boundary value problem arising from Example 2 of Jin and Levermore [7] as

$$\text{(6.4)} \qquad \mu_i \Phi_i'(x) + \sigma_T(x)\Phi_i(x) = \tfrac{1}{2} w(x)\sigma_T(x) \sum_{j=1}^{n} \omega_j \Phi_j(x), \qquad i = 1, \cdots, n,$$

$$\text{(6.5)} \qquad \Phi_i(0) = 5, \ \ i = \tfrac{1}{2}n + 1, \cdots, n, \qquad \Phi_i(11) = 0, \ \ i = 1, \cdots, \tfrac{1}{2}n.$$

Because of separation of the boundary conditions and the well-posed nature of this problem, we would expect exactly half of the fundamental modes to be nonincreasing and half to be nondecreasing across the entire interval. Our implementation of PSC indicates that this is indeed the case. No "spurious" dichotomy changes are flagged on any of the examples we tried.

Results for the two algorithms are given in Tables 6.4 through 6.7 for the cases $n = 20$ and $n = 10$. A box discretization was used with constant interval length and

Example 2. PSC: *Timings with $n = 20$ and number of intervals per processor fixed at* 200. *All times are in seconds.*

Number of processors	1	2	4	8	16
Time for initial reduction	2.18	2.22	2.21	2.19	2.19
Time for reduced system solution	.028	.044	.088	.113	.093
Total time	2.26	2.31	2.36	2.35	2.33

Example 2. SQR: *Timings with $n = 20$ and number of intervals per processor fixed at* 200. *All times are in seconds.*

Number of processors	1	2	4	8	16
Time for initial reduction	6.61	6.38	6.37	6.37	6.38
Time for reduced system solution	.016	.049	.140	.190	.206
Total time	6.72	6.53	6.61	6.66	6.68

Example 2. PSC: *Timings with $n = 10$ and number of intervals per processor fixed at* 200. *All times are in seconds.*

Number of processors	1	2	4	8	16	32	64	128
Time for initial reduction	.449	.450	.450	.450	.451	.451	.452	.452
Time for reduced system solution	.006	.009	.012	.034	.028	.041	.043	.034
Total time	.473	.477	.480	.502	.497	.510	.512	.504

Example 2. SQR: *Timings with $n = 10$ and number of intervals per processor fixed at* 200. *All times are in seconds.*

Number of processors	1	2	4	8	16	32	64	128
Time for initial reduction	1.27	1.27	1.27	1.27	1.27	1.26	1.26	1.27
Time for reduced system solution	.004	.010	.016	.031	.029	.034	.040	.054
Total time	1.30	1.31	1.32	1.33	1.32	1.33	1.33	1.35

a fixed number of intervals per processor. As in Example 1, the time taken to solve the reduced system tends to increase with the number of levels in the tree (though not smoothly), but high efficiency is obtained even on a large number of processors.

REFERENCES

[1] 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.

[2] U. M. Ascher and R. D. Russell, *Reformulation of boundary value problems into "standard" form*, SIAM Review, 23 (1981), pp. 238–254.

[3] H.-G. Bock, *Recent advances in parameter identification techniques for O.D.E.*, vol. 2 of Progress in Scientific Computing, Birkhauser, Boston, 1983, pp. 95–121.

[4] F. R. de Hoog and R. M. M. Mattheij, *An algorithm for solving multi-point boundary value problems*, Computing, 38 (1987), pp. 219–234.

[5] ———, *On the conditioning of multipoint and integral boundary value problems*, SIAM Journal of Mathematical Analysis and Applications, 20 (1989), pp. 200–214.

[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 2nd ed., 1989.

[7] S. Jin and D. Levermore, *The discrete-ordinate method in diffusive regimes*, Transport Theory and Statistical Physics, 20 (1991), pp. 413–439.

[8] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, 1974.

[9] 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.

[10] ———, *On boundary value problems for ordinary differential equations with parameters*, in Differential Equations, C. M. Dafermos et al., eds., Marcel Dekker, New York, 1989, pp. 481–489.

[11] S. J. Wright, *Stable parallel elimination for boundary value ODEs*, Preprint MCS–P229–0491, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, April 1991.

[12] ———, *Stable parallel algorithms for two-point boundary value problems*, SIAM Journal on Scientific and Statistical Computing, 13 (1992), pp. 742–764.