Domain Decomposition for Nonsymmetric Systems of Equations: Examples from Computational Fluid Dynamics

David E. Keyes*
William D. Gropp†

Abstract.
A block-preconditioned Krylov method which combines features of several previously developed techniques in domain decomposition and iterative methods for large sparse linear systems is described and applied to a few illustrative problems. The main motivation of the work is to examine the gracefulness of parallelization under the domain decomposition paradigm of the solution of systems of equations typical of finite-differenced fluid dynamical applications. Such systems lie outside of the realm of self-adjoint scalar elliptic equations for which most of the theory has been developed, and the present contribution is merely a first step in an attempt to approach them, raising several issues and settling none. However, results of tests run on an Encore Multimax with up to 16 processors show that even this first step has utility in the coarse-granularity parallelization of hydrocodes of practical importance.

1. Introduction

Interest in domain decomposition techniques for partial differential equations from a parallel computing perspective stems from their transformation of a large discrete problem defined on a spatial domain, irreducible in the matrix theory sense, into a fairly arbitrary and potentially large number of independent problems defined on simply-connected subdomains, at the cost of introducing a set of constraints at the interfaces of the subdomains. The interface equations contain all of the global coupling of the original problem; the subdomain problems are completely decoupled from each other once the interfacial degrees of freedom are assigned. The interfacial constraints, though complex in structure, involve lower-dimensional subsets of the unknowns of the original problem. With the development of sufficiently good approximations for the interface constraints, a preconditioned iterative scheme involving all of the unknowns of the problem can be constructed. The solution

*Department of Mechanical Engineering, Yale University, New Haven, CT 06520. The work of this author was supported in part by the National Science Foundation under contract number EET-8707109.
† Department of Computer Science, Yale University, New Haven, CT 06520. The work of this author was supported in part by the Office of Naval Research under contract N00014-86-K-0310 and the National Science Foundation under contract number DCR 8521451.

(or preconditioning) of the union of the subdomain problems is generally cheaper than the solution (or preconditioning) of the full domain problem due to their reduced size and bandwidth, in terms of which the appropriate operation counts may grow as badly as cubicly, depending upon the algorithm. This advantage can be traded off against the extra iterations needed due to the approximation of the interface equations. In the parallel context, additional advantages stemming from the independence and data locality of the subdomain problems enter into consideration.

In d-dimensional problems, the process of relegating the global coupling, inevitable at some level, to a set of lower-dimensional problems can, in principle, be recursively implemented d times. Thus, for example, a two-dimensional elliptic problem discretized with n subintervals on a side and decomposed into box-type subdomains with p subdomains on a side yields after two stages an irreducible system of $(p-1)^2$ equations for the points at the vertices of the boxes. This system is the Schur complement in the ambient (or global) matrix of the degrees of freedom defined at all other points [9]. Were the vertex degrees of freedom determined, $2p(p-1)$ decoupled sets of $(p-1)$ equations could be derived to yield the values at the points along the edges connecting them. Finally, given all of the interfacial degrees of freedom, p^2 decoupled sets of $(\frac{n}{p}-1)^2$ equations, simply sub-blocks of the original matrix, would yield the remaining values in the subdomain interiors. For general operators, the cost of deriving the exact lower-dimensional systems can greatly exceed the cost of direct banded Gaussian elimination on the original system. It follows that the success of domain decomposition hinges on the ability to efficiently approximate the lower-dimensional operators, by taking advantage of either known or "probed" structure.

Most published work to date on domain decomposition algorithms has concentrated on self-adjoint scalar elliptic equations, and several optimal algorithms are now known for this case, in the sense that the number of iterations required to solve the discretized PDE does not grow with the gridpoint density or the number of subdomains as these quantities are refined in proportion (see, *e.g.*, [5, 26]). These algorithms employ the conjugate gradient method, and are distinguished from each other primarily by the selection of solvers or preconditioners for the decoupled systems of equations for the subdomain interiors and for the coupled interface equation system. For certain constant coefficient problems, exact preconditioners can be obtained by means of Fourier analysis so that the iterations converge in a single step [4, 7]. Recently, Chan [6] has extended the class of problems for which good preconditioners are known to the scalar convective-diffusive case. However, problems involving several linearized convective-diffusive equations coupled to each other by source terms have received little attention in this context. The solution of such problems is an important computational kernel in implicit methods (for instance, Newton-like methods and linearized implicit time-stepping methods) commonly used for systems of nonlinear PDEs arising in science and engineering, and is often CPU-bound or memory-bound or both on the fastest and largest serial computers available. Furthermore, it is often the only computationally intensive part of such codes whose efficient parallelization is not straightforward, particularly when the distribution of data throughout the computer's memory hierarchy cannot be dictated exclusively by linear algebra considerations.

In this contribution, preconditionings of a "modified Schur complement" (MSC) type are applied to the solution of the large sparse linear systems arising at each stage of the application of Newton's method to a system of elliptic boundary value problems modeling a two-dimensional reacting fluid flow. The nomenclature derives from the fact that the preconditioner is built from blocks which are low-bandwidth approximations to actual Schur complements derived from various submatrices of the original global matrix. These low-bandwidth matrices are required to produce the same matrix-vector products as approximations to the true (dense) Schur complement when acting on a given set of trial vectors, and are obtainable by solving independent problems on subdomains. Since this type of requirement is sometimes imposed on incomplete factorizations, in which context the adjective "modified" has become solidly established, we employ it here as well. As a preconditioning technique for the interface equations, the modified Schur complement method exploits only the sparsity structure and clustering of large magnitudes around the diagonal of the actual operator, and no other properties like symmetry or constant coefficients. As a result, it performs suboptimally on many special problems, but generalizes with little difficulty to many other systems. In the absence of a general theory for the performance of this technique, experimental exploration of its convergence properties is required, and sample results are described herein (section 4), preceded by a description of the overall computational procedure (section 2), and its parallel implementation (section 3). We conclude in section 5 by listing some open issues in MSC-mediated domain decomposition.

2. Algorithmic Description

As sketched above, an iterative substructuring algorithm consists of an iterative procedure together with at least a two-level hierarchical preconditioner for the subdomain and interface systems. Our selection of a hybrid generalized minimum residual / incomplete LU-decomposition / modified Schur complement block algorithm is now motivated and described. The techniques woven into this hybrid are outlined in the following subsections. Fuller details are obtainable from the cited references.

2.1. Full Matrix Domain Decomposition

Domain decomposition enters our considerations at the level of the solution of a system of linear equations. In applications, this system is usually derived from the linearization of a nonlinear process, which gives the entire domain decomposition procedure the status of an inner iteration. To avoid any deeper nesting of iterations in applications, we iterate simultaneously on all of the unknowns in the linear system, in the sense that the subdomain problems are not individually iterated to convergence before their values are used to update the right-hand sides of the equations for the interfacial unknowns. This form of full matrix domain decomposition was advocated in [5] for problems in which no fast solver is known for the subdomain interior problems, and has been demonstrated therein to lead to an optimally convergent scheme for a class of self-adjoint strongly elliptic operators, provided (among other things) that spectrally equivalent subdomain preconditioners are employed.

This paradigm for domain decomposition is most easily illustrated in the decomposition into two strips of a rectangular region overlaid by a tensor-product grid. The single cut follows a line of gridpoints, which are ordered separately. For a 9-point operator on a grid with 16 interior subintervals in each direction (with Dirichlet boundary conditions eliminated) the resulting sparsity pattern for the operator A is indicated graphically in Fig. 1(a), and in matrix notation as follows:

$$A = \begin{pmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} . \qquad (1)$$

Here, A_{33} renders the coupling between the points on the interface itself.

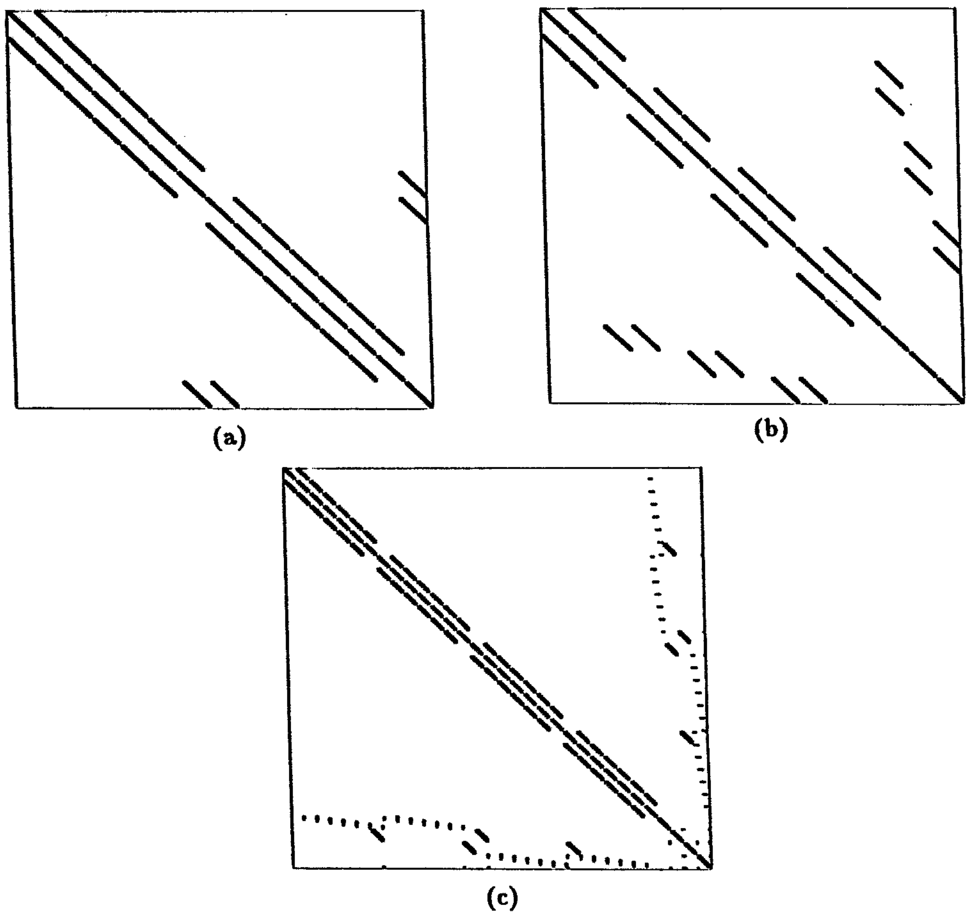

Figure 1: Sparsity patterns for three different decompositions of a rectangular region, using a 9-point finite-difference template: (a) two strips, one edge; (b) four strips, three edges; (c) four boxes, four edges, one vertex. Within each subdomain the gridpoints are ordered lexicographically.

The conformally partitioned preconditioning matrix we propose for A is

$$B = \begin{pmatrix} \tilde{A}_{11} & 0 & \tilde{A}_{13} \\ 0 & \tilde{A}_{22} & \tilde{A}_{23} \\ 0 & 0 & \tilde{C} \end{pmatrix}, \tag{2}$$

where $\tilde{C}$ approximates the Schur complement of A_{11} and A_{22} in A. The exact Schur complement, C, may be obtained from block-Gaussian elimination on A as:

$$C \equiv A_{33} - A_{31}A_{11}^{-1}A_{13} - A_{32}A_{22}^{-1}A_{23}. \tag{3}$$

The tilde-notation in the definition of B accommodates the replacement, if convenient, of the exact A_{ij} with approximations thereto. We assume throughout that the A_{ii} are invertible. (This is certainly a reasonable requirement for a discrete convective-diffusive operator.) Under this assumption, C is also invertible [9].

It has not been assumed above that A is symmetric. This provides the freedom to consider, without sacrifice of symmetry, a nonsymmetric B in (2) possessing instead the valuable property of block triangularity. Note that the inverse of B can be applied with one solve in each subdomain. Were A of the symmetric form

$$A = \begin{pmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{13}^T & A_{23}^T & A_{33} \end{pmatrix} , \tag{4}$$

we would have considered a symmetric B, in order to take advantage of the two-term recurrence relationship of the conjugate gradient iteration for the preconditioned system:

$$B = \begin{pmatrix} \tilde{A}_{11} & 0 & \tilde{A}_{13} \\ 0 & \tilde{A}_{22} & \tilde{A}_{23} \\ \tilde{A}_{13}^T & \tilde{A}_{23}^T & \tilde{C} + \sum \tilde{A}_{i3}^T \tilde{A}_{ii}^{-1} \tilde{A}_{i3} \end{pmatrix} . \tag{5}$$

One application of the inverse of this symmetric B requires two solves in each subdomain [5], twice the work of the block triangular B.

2.2. Generalized Minimum Residual Method

Any algorithm intended for use in fluid dynamical applications must be robust with respect both to asymmetry, to allow for the presence of convective terms, and indefiniteness, to allow for the presence of linearized source terms whose coefficients oppose the algebraic sign of the diagonal term of the discrete convective-diffusive operator. (The latter can be particularly important in the modeling of chemically reacting flows in which at any point in the flow field some species may be created while others are consumed. The source terms in the transport equations for electrons and holes frequently employed in semiconductor device simulation can also give rise to indefiniteness.) The generalized minimum residual method (GMR) [21] is one such algorithm with which the authors have experience. Chebyshev iteration [17] would be an alternative suitable for cases in which the indefiniteness can be controlled (for instance, by the addition of a transient term to the continuous operator, which adds a term inversely proportional to some sufficiently small time step and of the correct sign to the diagonal of the discrete operator). All of our results to date employ GMR; however, we intend to experiment with adaptive Chebyshev iteration in the future because of its short recurrence relation.

Given a system of equations, $Mx = f$, M nonsingular, and an initial iterate, x_0, with initial residual, $r_0 = f - Mx_0$, GMR computes the solution x from finding $z \in K_m$ such that

$$(r_0 - Mz, v) = 0 \ ,$$

for all $v \in L_m$, and setting $x = x_0 + z$, where K_m and L_m are Krylov spaces based on r_0:

$$K_m \equiv \text{span}\{r_0, Mr_0, \ldots, M^{m-1}r_0\},$$

$$L_m \equiv \text{span}\{Mr_0, M^2r_0, \ldots, M^m r_0\}.$$

The solution x computed after m steps of GMR minimizes $||r||_2$ in the affine space $x_0 + K_m$. In a practical algorithm, an orthogonal basis for K_m is built up by means of a Gram-Schmidt or Householder process, which obviates the necessity of working with the normal equations. Suitable computer implementations of GMR have been given in [21] and [24], of which we use the former. Among the desirable properties of GMR are: (1) the only reference to M is in form of matrix-vector products, (2) the only matrix to be "inverted" is usually of order $m \ll \dim(M)$, (3) it cannot break down (in exact arithmetic) short of delivering the solution even for nonsymmetric systems with indefinite symmetric part, (4) it requires less storage and fewer operations per step than the mathematically equivalent GCR and ORTHODIR algorithms, and (5) the 2-norm of the residual is non-increasing and can be monitored without constructing intermediate solution iterates. The main disadvantage of GMR is the lack of a bounded recurrence relation, which causes the operation count and storage requirements to grow quadratically and linearly, respectively, in the iteration index. In many applications, restarting GMR after a predetermined number of steps successfully deals with this problem, but restarted GMR can also fail through stagnation.

2.3. Incomplete LU Subdomain Preconditioning

GMR is often uneconomical when left to act by itself on a general reaction-convection-diffusion operator. In an effort to control the work and storage required by GMR when A has a widely spread spectrum, we precondition the iterations by taking M in the formulae above to be AB^{-1}, for the A and B given in §2.1. This is "right" preconditioning, which first solves $M\tilde{x} = f$ for $\tilde{x}$, then $Bx = \tilde{x}$ for x. We adopt right over left preconditioning because in the latter the matrix B enters into the GMR residual convergence criterion in a direct way, making convergence comparisons between different preconditioning techniques difficult.

Approximate factorizations of the original matrix into triangular matrices, such as incomplete LU-decomposition (ILU) [18], are useful general purpose preconditioners for GMR and other Krylov methods. However, such factorizations can be bottlenecks in parallel implementations, because of the sequential nature of triangular factorizations and solves. Though wavefront-based or red-black reorderings of the standard sequential operations can alleviate this problem [20] in the context of sparse banded matrices, domain-decomposition approaches side-step it altogether by applying ILU within subdomains only.

In the present examples we employ the simplest of techniques appropriate to nonsymmetric systems, a Crout-Doolittle ILU(0), where the zero indicates that no fill-in outside the original sparsity pattern of A is allowed (see [25]). Thus, by its definition, the remainder matrix $R \equiv LU - A$ has (in the nine-point case) four nonzero diagonals, one on either side of the tridiagonal cluster about the main diagonal and one just inside each of the exterior tridiagonal clusters.

In a procedural language aided by compass-point subscript notation we have the following algorithm for a 9-point stencil in two dimensions:

$$\begin{aligned}
u_0^i &\leftarrow a_0^i - l_{SW}^i u_{NE}^{i-n-1} - l_S^i u_N^{i-n} - l_{SE}^i u_{NW}^{i-n+1} - l_W^i u_E^{i-1} \\
u_E^i &\leftarrow a_E^i - l_S^i u_{NE}^{i-n} - l_{SE}^i u_N^{i-n+1} \\
u_{NW}^i &\leftarrow a_{NW}^i - l_W^i u_N^{i-1} \\
u_N^i &\leftarrow a_N^i - l_W^i u_{NE}^{i-1} \\
u_{NE}^i &\leftarrow a_{NE}^i \\
l_W^{i+1} &\leftarrow [a_W^{i+1} - l_{SW}^{i+1} u_N^{i-n} - l_S^{i+1} u_{NW}^{i-n+1}][u_0^i]^{-1} \\
l_{SE}^{i+n-1} &\leftarrow [a_{SE}^{i+n-1} - l_S^{i+n-1} u_E^{i-1}][u_0^i]^{-1} \\
l_S^{i+n} &\leftarrow [a_S^{i+n} - l_{SW}^{i+n} u_E^{i-1}][u_0^i]^{-1} \\
l_{SW}^{i+n+1} &\leftarrow [a_{SW}^{i+n+1}][u_0^i]^{-1}
\end{aligned} \tag{6}$$

Here a_0^i is the diagonal term in the ith row, a_E^i is the term in the first superdiagonal of the ith row, which corresponds in the natural ordering to the point due east of the diagonal entry, etc. There are n interior gridpoints in the most rapidly ordered direction. The entries $u_{()}^i$ and $l_{()}^i$ of the sparse upper and lower triangular factors overwrite partial rows and columns, respectively, of the original matrix as the factorization proceeds down the main diagonal. (l_0^i need not be computed because it is simply the identity.) Higher order factorizations, denoted ILU(k), $k > 0$, are possible in which more nonzero diagonals are permitted to accommodate the fill-in in U and L, relative to A. For $k = n - 2$ with the

9-point stencil (or $k = n - 1$ with the 5-point stencil), the band fills in completely and the factorization becomes exact.

2.4. Modified Schur Complement Interface Preconditioning

For the interface equations, we use the modified Schur complement (MSC) technique. This was proposed in [7] and implemented and compared with several other preconditioners on symmetric problems in [14]. Like the ILU technique, MSC allows for a variable number of nonzero diagonals. In the two-dimensional two-subdomain example (1), we define MSC(k) by using for the matrix $\tilde{C}$ in (2) the following banded matrix:

$$\tilde{C}_k = \tilde{A}_{33} - E_k ,$$

where E_k is banded of semi-bandwidth k which satisfies as accurately as possible

$$E_k v_l = \sum_i (\tilde{A}_{3i} \tilde{A}_{ii}^{-1} \tilde{A}_{i3}) v_l \tag{7}$$

for a set of vectors v_l, $l = 1, 2, \dots, L$.

For a nonsymmetric scalar system of equations, we set $L = 2k + 1$ and use for the respective E_k the vectors:

$$\begin{aligned}
E_0 : v_1 &= [1,1,1,1,1,1,\dots]^T \\
E_1 : v_1 &= [1,0,0,1,0,0,\dots]^T \\
v_2 &= [0,1,0,0,1,0,\dots]^T \\
v_3 &= [0,0,1,0,0,1,\dots]^T \\
E_2 : v_1 &= [1,0,0,0,0,1,\dots]^T \\
v_2 &= [0,1,0,0,0,0,\dots]^T \\
v_3 &= [0,0,1,0,0,0,\dots]^T \\
v_4 &= [0,0,0,1,0,0,\dots]^T \\
v_5 &= [0,0,0,0,1,0,\dots]^T \\
&\vdots
\end{aligned} \tag{8}$$

For a symmetric scalar system of equations, we set $L = k + 1$ and use instead:

$$\begin{aligned}
E_0 : v_1 &= [1,1,1,1,1,1,\dots]^T \\
E_1 : v_1 &= [1,0,1,0,1,0,\dots]^T \\
v_2 &= [0,1,0,1,0,1,\dots]^T \\
E_2 : v_1 &= [1,0,0,1,0,0,\dots]^T \\
v_2 &= [0,1,0,0,1,0,\dots]^T \\
v_3 &= [0,0,1,0,0,1,\dots]^T \\
&\vdots
\end{aligned} \tag{9}$$

In the nonsymmetric case, there are $(2k + 1)n - k(k + 1)$ distinct elements in E_k of dimension n, and $(2k + 1)n$ scalar equations in (7). Therefore, only the $k = 0$ (simple row-sum preserving) case is well defined. $(2k + 1)n - k(k + 1)$ of the equations in (7) explicitly assign individual elements of E_k. The remaining $k(k+1)$ equations, for $k>0$, involve none of the elements of E_k at all, but require a certain sum of elements from the matrix on the right-hand side of (7), each of which is at least $k+1$ diagonals away from the main diagonal, to vanish. This overdetermination is inconsistent, but not fatally so for a preconditioner, particularly if the Schur complement matrix being approximated has terms which decay rapidly away from the diagonal. It costs $2k+1$ solves on each subdomain to compute the right-hand side of (7).

In the symmetric case, there are $(k+1)n - k(k+1)/2$ distinct elements in E_k of dimension n and $(k+1)n$ scalar equations in (7). In this case, however, the overdetermination is consistent. The diagonal elements of E_k can be read off from explicit assignments and the remaining elements can be obtained in $O(kn)$ operations. It costs $k+1$ solves on each subdomain to compute the right-hand side of (7).

In either case, setting $k=n-1$ determines the Schur complement exactly (assuming that the tilde-quantities in (7) are identical to their non-tilde counterparts), in which case the overall algorithm converges in one iteration. This is, in effect, the approach used by Przemieniecki in [19], and requires as many subdomain solves as there are degrees of freedom on the interface. As a means of obtaining a specified finite level of precision in the final result (commensurate, for example, with the discretization error), taking k to be $O(n)$ is inefficient, and particularly so if the subdomain solves are themselves not exact so that more than one iteration is needed. Between the extremes of many cheap iterations and few expensive iterations determined by the index k in each of ILU(k) and MSC(k) will be an optimal trade-off. In subsequent sections the simple gridpoint-scale zero patterns in these v_l will need extensions to both coarser (§2.4.1) and finer (§2.5) scales.

2.4.1. Extension of MSC to Multiple Interfaces

The descriptions of MSC(k) above are for a simple interface between two subdomains, as depicted in Fig. 1(a). Generalizations to the compound interface cases shown in Fig. 1(b) (three edges) and Fig. 1(c) (four edges, with vertex) and finer decompositions can take a variety of forms. Their development is aided by a heuristic rather than formal approach to approximating the appropriate compound-interface generalization of the Schur complement (3), recognizing it to be, essentially, a discrete Green's function.

We note that the element C_{ij} represents the influence of the data at interfacial node j (source point) on the discrete residual at interfacial node i (field point). Varying each interface point separately in turn would enable filling in C column-by-column. In diffusive problems, we would expect this internodal influence to decay rapidly with physical separation. In mixed convective-diffusive problems, we would expect dependencies of even shorter range on the values at "downwind" nodes, and of somewhat longer range on the values at "upwind" nodes. Depending upon the magnitude of these influences we might want, in the interest of economy and efficient parallelism, to set large off-diagonal blocks of C corresponding to sufficiently distantly coupled degrees of freedom to zero, and to determine the remaining (assumed non-neglible) blocks by varying large numbers of source points simultaneously. This is analogous to the Curtis-Powell-Reid [10] technique for efficient sparse Jacobian estimation using vector function evaluations, except that we are prepared, in general, to accept much looser restrictions on which columns may be treated as corresponding to unrelated degrees of freedom and thus be evaluated simultaneously.

In the extreme limit of $k=0$, we attempt to probe all columns of C simultaneously. As k grows, the resources for resolving more of the structure of C can be invested in different ways. This is done by the selection of the v_l in (7). The v_l listed above are appropriate in a purely diffusive problem with a spatially uniform diffusion coefficient and isolated interfaces (assuming that all the nodes on a given interface are ordered consecutively). By spreading out the active source points as evenly as possible, these v_l put a premium on resolving the influence of nearest neighbors *along* an interface. In a multiple-interface problem in which the subdomains possess high aspect ratio, it may be a better investment to isolate physically nearby interfacial degrees of freedom belonging to different edges than to isolate degrees of freedom distantly separated on the same edge. For the v_l above, taking k greater than the narrowest discrete dimension of the high-aspect-ratio subdomains would give rapidly diminishing returns, with a similar problem occuring for boxwise decompositions near the vertices. In such cases, we recommend assigning parity to the decomposition-defining cuts (in each dimension) and evaluating MSC blocks corresponding to odd and even parity in separate stages. In each stage, only one set of cuts bordering each subdomain is active; the elements of the v_l corresponding to all others are identically zero. For stripwise decompositions, this requires twice as many subdomain solves to compute $\tilde{C}$. For boxwise decompositions, four times the original preprocessing work is required.

2.4.2. Extension of MSC to Higher Levels of Complementation

The sparsity structure of the discrete operator A for the simplest decomposition admitting a two-level complementation is furnished in Fig. 1(c). The ambient matrix

$$A = \begin{pmatrix} A_{11} & 0 & 0 & 0 & A_{15} \\ 0 & A_{22} & 0 & 0 & A_{25} \\ 0 & 0 & A_{33} & 0 & A_{35} \\ 0 & 0 & 0 & A_{44} & A_{45} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} \end{pmatrix}, \qquad (10)$$

where A_{55} renders the coupling between all the interfacial points (the union of the edge and vertex unknowns), is a bordered block diagonal matrix.

Eliminating A_{11} through A_{44} leaves a capacitance system which is *not* sparse in general, a situation which is fundamentally very different from that encountered in the ambient matrix. This poses problems in the parallel context, since it is desirable to construct a block triangular preconditioner for it in which each edge is handled independently.

Fortunately, cases are known in which bordered block diagonal approximations to this first Schur complement make acceptable preconditioners. The (symmetric) purely diffusive case is covered in [5], and the convectively-dominated case is discussed below in the absence of recirculation. Especially in such cases, though also in others, it may be worthwhile to recursively employ MSC techniques to approximate the Schur complement of the edge degrees of freedom in the matrix which approximates the first complement, and use the result to precondition the vertex equation system (which consists of a single degree of freedom in the example of Fig. 1(c)). It should be emphasized that the practical utility of higher-level MSC techniques has yet to be demonstrated. However, the fruitfulness of solving a globally coupled vertex system in the purely diffusive context [5], warrants attempted generalizations to other operators. Multigrid technology provides another, alternative path. The main advantage of the higher-level MSC approach will likely be the same as at the first level: it provides a path to constructing a preconditioner which becomes arbitrarily accurate as the bandwidth k is increased at all levels, independent of special operator features. Against this advantage must be traded off the expense of approximately solving exponentially more subdomain problems at each level of complementation.

2.5. Block-structured GMR/ILU/MSC domain decomposition

In most fluid dynamical applications there are several fields defined at each point in the domain, including the momenta and pressure (alternatively the streamfunction and vorticity), and possibly species concentrations, temperature, density, etc. The several unknowns at a single gridpoint couple strongly with one another, not only directly through source terms, but also indirectly through material-dependent transport properties in multicomponent mixtures. In the ordering of these fields into a discrete vector of unknowns, it is natural to preserve the locality of this coupling by grouping the r unknowns defined at each point together. This brings about a block-structured linearized operator in which the $r \times r$ blocks are dense, but distributed sparsely, typically in a 5- or 9-diagonal structure for two-dimensional problems or a 7- or 27-diagonal structure in three dimensions.

The ILU and MSC techniques can be generalized to accommodate multicomponent problems. In §2.3 the $a^i_{()}$, $u^i_{()}$ and $l^i_{()}$ must be reinterpreted as $r \times r$ blocks and the ones in (8) replaced with each of the r unit vectors in sequence, *e.g.*,

$$\begin{aligned} E_0 : v_{1,1} &= [e_1, e_1, e_1, e_1, e_1, e_1, \ldots]^T \\ v_{1,2} &= [e_2, e_2, e_2, e_2, e_2, e_2, \ldots]^T \\ &\vdots \\ v_{1,r} &= [e_r, e_r, e_r, e_r, e_r, e_r, \ldots]^T \end{aligned} \tag{11}$$

To resolve the intra-point coupling in the MSC method requires $(2k+1)r$ subdomain solves in the nonsymmetric case. Block-point preconditioning has been found superior in the single-domain context to the componentwise ILU schemes with the same storage requirements on the convective-diffusive-reactive operators considered in §4. The virtues of block-line preconditioning in the $r = 1$ case have also been explored in [8] and [23].

3. Parallel Implementation

There are potentially three penalties to be paid in distributing the GMR/ILU/MSC solution algorithm over an array of independent processors: synchronization overhead, communication overhead, and degradation of convergence. These penalties are measured indirectly through the speedup and efficiency figures-of-merit of a parallel implementation. The speedup is the ratio of the uniprocessor execution time of a given algorithm to that of the multiprocessor execution of the same algorithm. The efficiency is usually defined as the speedup divided by the number of processors. For many algorithms, these definitions are unnatural in the sense that one would never use the same algorithm in both uniprocessor and multiprocessor environments. (Usually *better* uniprocessor algorithms exist, so the parallel efficiency as defined above is inflated relative to its advantage.) We adopt measures in which the execution times are obtained from the most *natural* algorithm for each environment, namely, given p processors we employ exactly p subdomains.

The synchronization penalty arises if the processors have dependencies which force them them to wait for data which are not yet available. Even if the processors are programmed homogeneously this can happen at convergence checkpoints, for instance, if they have unequal amounts of work to do, or at points where reduction to a scalar of data distributed over all processors is required. The amount of work to be done in a processor is a function of the number of gridpoints in the subdomain assigned to it and the stencil of the discrete equations to be enforced at those gridpoints. The relative number of boundary gridpoints (which require somewhat less computational work than interior ones) decreases as the mesh is refined, and though their distribution between the processors becomes more uneven, only a small number of processors are thus periodically idled. If the gridpoints are allocated to the processors as evenly as possible within shape and contiguity constraints (which is *not* accomplished in our preliminary examples to follow) synchronization delays can be made relatively unimportant.

The communication penalty is the time spent gaining access to shared data even after it becomes available. The significance of this penalty depends on the amount of data to be shared, on its routing between memory and processors, on the amount of arithmetic which the algorithm must perform between fetches and writes, and on the communication-to-computation speed ratios of the hardware in question.

In an effort to increase the number and length of the independent threads which comprise a parallel computation, global coupling may be reduced in ways that degrade the convergence rate of the algorithm. In the context of domain decomposition, this penalty may arise as the ratio of interface to interior degrees of freedom is increased in refinements of the decomposition, since the approximations required to form diagonal preconditioner blocks for the former are often more severe.

In view of the above considerations as well as programming convenience, our parallel implementation consists of decomposing the logical tensor product computational domain into logically congruent strips or boxes of contiguous unknowns in two dimensions, (with the obvious generalizations in three dimensions) mapping these subdomains onto a network of processors, and programming the processors homogeneously. In this paper, our principal interest is in convergence rates, so we report on a bus-connected shared memory machine only: an Encore Multimax 320. The dependent variable arrays are placed in the shared memory and each processor is confined to roam over subranges of the array indices. The timings in the tables to follow include the parallel generation of the ILU and MSC blocks and the entire GMR iteration. To reduce the number of synchronization points inside each GMR iteration, a small QR factorization from which the coefficients of the Krylov basis vectors in the solution vector are derived is carried out redundantly in each processor. An analogous consideration led to the redundant solution of the equations for the vertex degrees of freedom in the parallel domain decomposition method described in [13]. Certain pre- and post-processing tasks, like the coefficient generation of the original operator and some spectral analysis, are done in serial and not included in the timings below.

4. Results for Model Problems

This section contains numerical results that display a few of the possibilities of MSC preconditioning, and, more generally, of the hybrid GMR/ILU/MSC algorithm. In the solution of linear systems arising from finite-differenced systems of conservation equations of the form

$$\frac{\partial u_r}{\partial t} + \vec{c}(\vec{u}) \cdot \nabla u_r - \nabla \cdot D_r(\vec{u}) \nabla u_r = f_r(\vec{u})$$

for $r = 1, \ldots, R$, combinations of the first, third, and fourth terms can be handled well with known methods [7], when $R = 1$. Comparisons on purely diffusive problems of MSC(k) for $k = 1, 2, 3$ with the optimal preconditioners may be found in [14]. Therefore, the cases $|\vec{c}| > 0$ and $R > 1$ are of particular interest. In the preliminary results contained herein, only stripwise decompositions (one level of complementation) are considered, and $k = 0$ in both MSC(k) and ILU(k).

4.1. Scalar Convection-Diffusion Problems

In the limit of unbounded mesh refinement and bounded convection velocity c, the nonsymmetric first term in the discrete operator for

$$cu_y - D\nabla^2 u = f \tag{12}$$

will be dominated by the diffusive term, and the optimal preconditioners such as the square root of the one-dimensional Laplacian (denoted $K^{1/2}$ in [14]) will be asymptotically superior to any MSC preconditioner. However, much practical CFD computation occurs in the opposite singularly perturbed limit. Especially in the early, coarse-grid stages of an adaptive grid calculation it is important to have methods which support the artificially diffusive upwind-differencing of the convective term.

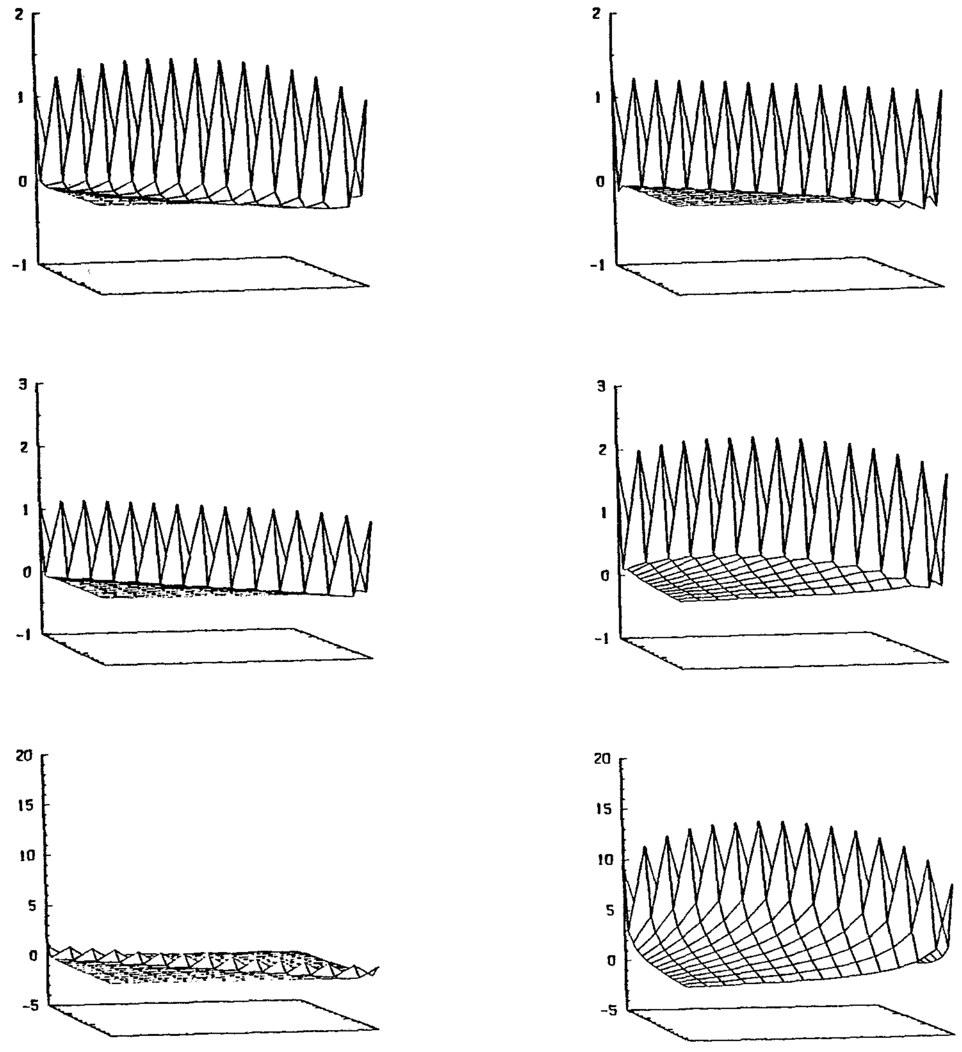

Figure 2: Element plots of the preconditioned capacitance system $\tilde{C}^{-1}C$ for the convective-diffusive operator (12) on the unit square with Dirichlet boundary conditions at $\mathrm{Re}_c = 0$ (top), $\mathrm{Re}_c = 2$ (middle), and $\mathrm{Re}_c = 20$ (bottom), using the modified Schur complement (left) and root-one-dimensional Laplacian (right). The horizontal axes are the row and column indices. Note that the vertical scale is different in each row of plots.

Fig. 2 consists of a set of plots of the matrix elements of the preconditioned interfacial system for the decomposition of Fig. 1(a), for the convective-diffusive operator of (12). To eliminate any interaction with ILU, the subdomain solves are carried out exactly in constructing the capacitance matrix. Two preconditioners, $\tilde{C}_D = K^{1/2}$ and $\tilde{C}_M$ = MSC(0), are considered at three cell Reynolds numbers, $\mathrm{Re}_c \equiv |c|h/D = 0$, 2, and 20, respectively. For a good preconditioner, $\tilde{C}^{-1}C$ should be as close as possible to the identity. It is clear that the "straw man" $\tilde{C}_D$ fails as Re_c departs from 0, while $\tilde{C}_M$ improves. Another means of visualizing this improvement is the plot of the capacitance spectra in Fig. 3.The clustering of the eigenvalues near unity as Re_c becomes large is due to the easily physically visualizable fact that at least one of the factors in each of the triple products in equation (3) becomes small relative to the diagonal term A_{33} in this limit. High Re_c parabolizes the flow, making the matrices A_{32} and A_{13}, which transmit information upwind when $c > 0$, negligible. The orthogonality of the flow direction and the interface is not essential to this argument. For high enough Re_c, at least one in each pair of (A_{13}, A_{31}) and (A_{23}, A_{32}) is negligible compared to A_{33} at any flow orientation, including alignment with the interface. However, it is essential that the interface not cut a zone of fluid recirculation.

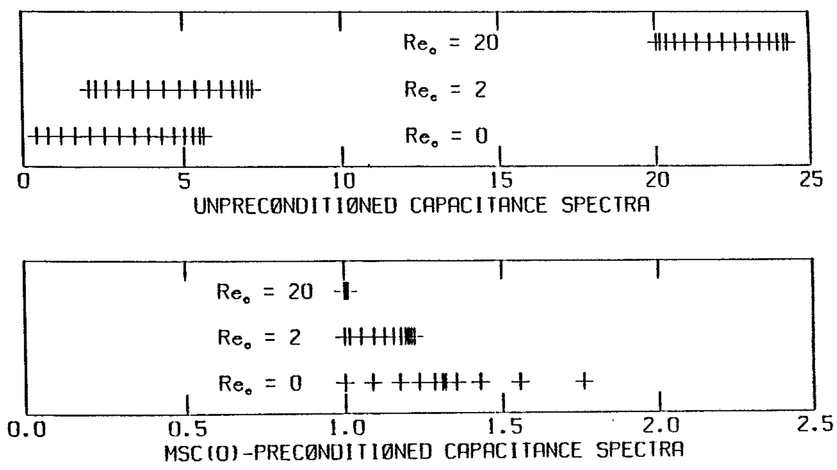

Figure 3: Spectra of the exact capacitance matrix for the convective-diffusive operator (above) and of the MSC(0)-preconditioned capacitance system (below), for three different cell Reynolds numbers.

The performance of MSC(0) in conjunction with approximate subdomain solves, is given in Table 1 for two different convective directions. (12) is the "vertical" convection case; for the "horizontal" we replace the first term with cu_x. Neumann conditions are used at the outflow boundary in either case and Dirichlet elsewhere on the perimeter of a unit square. All boundary degrees of freedom are incorporated into the matrix; none are eliminated *a priori*. Speedups of 5 to 10 are obtained on the largest problems using 16 processors. Note that in one convective direction the iteration count degrades at high aspect ratio, while in the other it actually improves slightly. As suggested by

		Vertical Convection			Horizontal Convection		
p	n	I	T	e	I	T	e
1	17	8	4.8	1.00	10	5.9	1.00
2	17	9	3.3	.73	11	3.9	.76
4	17	11	2.4	.50	11	2.6	.57
1	33	11	23.9	1.00	14	30.5	1.00
2	33	12	15.3	.78	15	19.0	.80
4	33	14	10.0	.60	15	10.7	.71
8	33	16	7.0	.43	15	6.5	.59
1	65	15	127.	1.00	21	186.	1.00
2	65	16	76.7	.83	22	109.	.85
4	65	17	43.1	.74	22	57.4	.81
8	65	20	28.7	.55	22	32.7	.71
16	65	26	24.7	.32	21	17.8	.65

Table 1: Iteration count I, CPU time T, and efficiency e for the convective-diffusive problem (12) at $\mathrm{Re}_c = 2.0$ with horizontal strips and either vertical convection or horizontal convection, as a function of number of processors p and spatial resolution n.

the different iteration counts in the undecomposed case $p = 1$, this anisotropy may be due as much to the difference in angle between the convection direction and that of the most rapid ordering of the unknowns in the ILU subdomain blocks (always horizontal) as to any interface orientation effect. Different flow orientations, interface orientations, and ILU orderings, leave a multitude of primitive combinations to be investigated. Note that neither combination considered here is superior to the other at all granularities of the decomposition. The relatively greater efficiency of the aligned case at high p is due to its poorer absolute $p = 1$ performance (compare the CPU times of the $n = 65$ data).

4.2. Vector Source-Diffusion Problems

As an example of a multicomponent problem with coupling through source terms, we consider a compact discretization of a sixth-order operator by means of three second-order operators. To be specific, we test

$$\begin{aligned} -\nabla^2 u_1 &= f \\ -\nabla^2 u_2 &= u_1\,, \\ -\nabla^2 u_3 &= u_2 \end{aligned} \tag{13}$$

on a unit square with Dirichlet boundary conditions for all components. This introduces bidiagonal 3×3 diagonal blocks which the hybrid algorithm regards as fully dense. As with scalar Poisson problems, preconditioning techniques superior to MSC can be devised for such systems by exploiting their special structure.

Results are shown in Table 2 for the same problem decomposed into either horizontal or vertical strips. Speedups close to 9 are achieved on 16 processors. Differences in performance between each pair of cases in the same row of the table can be induced only by the ordering of the ILU preconditioning, which is fastest along the interface direction for horizontal strips, and fastest in the normal direction for vertical strips. Non-monotonic iteration count behavior characterizes both orientations; in fact, the decrease in iterations in going from 2 to 4 subdomains in the $n = 65$ case contributes to superlinear relative speedup (the execution time is better than halved).

		Horizontal Strips			Vertical Strips		
p	n	I	T	e	I	T	e
1	17	11	18.4	1.00	11	18.4	1.00
2	17	12	12.9	.71	12	13.0	.70
4	17	13	8.1	.57	12	7.6	.60
1	33	18	109.	1.00	18	109.	1.00
2	33	22	78.2	.70	22	78.8	.69
4	33	20	38.9	.70	20	39.5	.69
8	33	23	25.1	.54	21	24.0	.57
1	65	33	818.	1.00	33	818.	1.00
2	65	40	564.	.73	40	567.	.72
4	65	37	267.	.77	37	271.	.75
8	65	37	145.	.71	36	145.	.71
16	65	40	96.1	.53	36	88.5	.58

Table 2: Iteration count I, CPU time T, and efficiency e for the coupled Poisson problem (13) with horizontal strips or vertical strips, as a function of number of processors p and spatial resolution n.

4.3. Vector Reaction-Convection-Diffusion Problems

We conclude with examples of Jacobian matrices from a real problem describing a two-dimensional axisymmetric methane-air laminar diffusion flame under the flamesheet approximation for the chemical kinetics. The flamesheet exploits a large Damkohler number (ratio of diffusion to reaction time scales) by replacing a reaction zone of finite thickness with an interface (of unknown location) across which gradients of the temperature and species are discontinuous. The interface subdivides the physical domain Ω into an oxidizer-free zone Ω_F and a fuel-free zone Ω_O, in either of which the full composition and thermodynamic state of the gas mixture can be recovered from a single conserved scalar. The flamesheet is an economical means of computing initial iterates for detailed kinetics calculations [15], since it requires just three components per gridpoint (a variable density Stokes streamfunction ψ, the vorticity ω, and the conserved scalar S) instead of the dozens that would be required with full chemistry. The continuous system is of the following form:

Streamfunction and Vorticity Definitions:

$$\rho r v_r = -\frac{\partial\psi}{\partial z}, \quad \rho r v_z = \frac{\partial\psi}{\partial r}, \quad \omega = \frac{\partial v_r}{\partial z} - \frac{\partial v_z}{\partial r}$$

Streamfunction Equation:

$$\frac{\partial}{\partial z}\left(\frac{1}{r\rho}\frac{\partial\psi}{\partial z}\right) + \frac{\partial}{\partial r}\left(\frac{1}{r\rho}\frac{\partial\psi}{\partial r}\right) + \omega = 0$$

Vorticity Equation:

$$r^2\left[\frac{\partial}{\partial z}\left(\frac{\omega}{r}\frac{\partial\psi}{\partial r}\right) - \frac{\partial}{\partial r}\left(\frac{\omega}{r}\frac{\partial\psi}{\partial z}\right)\right] - \frac{\partial}{\partial r}\left(r^3\frac{\partial}{\partial r}\left(\frac{\mu}{r}\omega\right)\right) - \frac{\partial}{\partial z}\left(r^3\frac{\partial}{\partial z}\left(\frac{\mu}{r}\omega\right)\right) +$$

$$r^2 g\frac{\partial\rho}{\partial r} + r^2\left[\frac{\partial}{\partial r}\left(\frac{v_r^2+v_z^2}{2}\right)\frac{\partial\rho}{\partial z} - \frac{\partial}{\partial z}\left(\frac{v_r^2+v_z^2}{2}\right)\frac{\partial\rho}{\partial r}\right] = 0$$

Conserved Scalar Equation:

$$\frac{\partial}{\partial z}\left(S\frac{\partial\psi}{\partial r}\right)-\frac{\partial}{\partial r}\left(S\frac{\partial\psi}{\partial z}\right)-\frac{\partial}{\partial r}\left(r\rho D\frac{\partial S}{\partial r}\right)-\frac{\partial}{\partial z}\left(r\rho D\frac{\partial S}{\partial z}\right)=0$$

State Relations:

$$\rho(\vec{x})=\left\{\begin{array}{ll}\rho_F(S), & \vec{x}\in\Omega_F\\ \rho_O(S), & \vec{x}\in\Omega_O\end{array}\right\}$$

$$\mu(\vec{x})=\left\{\begin{array}{ll}\mu_F(S), & \vec{x}\in\Omega_F\\ \mu_O(S), & \vec{x}\in\Omega_O\end{array}\right\}$$

$$D(\vec{x})=\left\{\begin{array}{ll}D_F(S), & \vec{x}\in\Omega_F\\ D_O(S), & \vec{x}\in\Omega_O\end{array}\right\}$$

In practice (see *e.g.*, [22, 16]), these equations are solved by a modified Newton method and a pseudo-transient continuation scheme (in which the time-step is adaptively chosen, becoming infinite asymptotically to recover the quadratic convergence of Newton's method) on an adaptively refined (and severely nonuniform) grid.

p	n	I	T	e
1	17	9	18.3	1.00
2	17	13	16.1	.57
4	17	18	12.5	.37
1	33	14	100.	1.00
2	33	17	73.3	.68
4	33	20	47.3	.53
8	33	22	29.1	.43
1	65	18	507.	1.00
2	65	18	300.	.85
4	65	18	158.	.80
8	65	18	87.0	.73
16	65	17	52.2	.61

Table 3: Iteration count I, CPU time T, and efficiency e for Jacobians from the flamesheet problem with horizontal strips at various stages of spatial resolution n, as a function of number of processors p.

Table 3 reports results on Jacobians drawn from different stages of a calculation in which an initially adaptively selected grid was constrained to be refined by factors of two to avoid load-balancing considerations. There is no relation between problem difficulty at the three resolutions chosen. In fact, the first two are near the domain of convergence of Newton's method on their respective grids (near infinite time step) while the last has a large diagonal contribution from the pseudo-transient term. The ten-fold speedups on large problems would lead to worthwhile improvements in turnaround times on detailed kinetics problems, in which several supercomputer CPU hours can be spent on linear algebra alone. Though linear algebra does not necessarily account for the dominant share of the total CPU time in such models, it is the only part whose parallelization is non-trivial. Furthermore, linear solvers of simple relaxation type *do* contribute to the leading order in the communication overhead in parallelization implementations developed to date [16]. Domain-decomposed solvers are capable of doing more arithmetic between neighbor-neighbor interchanges and global convergence checks, at least up to moderate decomposition granularities.

5. Conclusions and Directions for Further Research

Included in the GMR/ILU/MSC framework is a family of schemes governed by bandwidth parameters which lie between the extremes of decoupled block-diagonal preconditioned GMR and domain-decomposed direct elimination. Though surpassed in efficiency by known methods in several contexts, they provide a means for the parallel solution of rather general linear systems. Improvements should be sought, however, in several areas:

1. *Tuning of order of approximation between levels.* It is clear that each level in the MSC hierarchy is limited in its attainable preconditioning ability by the levels beneath it from which the right-hand sides of (7) are computed. There would clearly be vanishing returns in cranking up k indefinitely in MSC while leaving it fixed in ILU. Further work is needed to establish theoretically on model problems and experimentally on representative real problems how the different tuning parameters should be coordinated.
2. *Locally operator-adaptive approximation.* In problems with different physical properties on different subdomains the requirements on the preconditioning blocks at the same level in the hierarchy may vary from subdomain to subdomain. Depending on a locally-averaged Reynolds number, for instance, a fast Poisson solver might be a chosen over ILU on some subdomain. Alternatively, the k in ILU(k) might be chosen differently within different subdomains. A mature technique should exploit such versatility, perhaps even adaptively, taking into consideration a load-imbalance penalty function.
3. *Other approximate Schur complements.* MSC is only one means of obtaining a compact approximation of the Schur complement. In problems with simplifying structure, other compact approximations may exist. For instance, noting the near translation independence along an interface in a Poisson problem, Golub and Mayers proposed a Toeplitz approximation to C in [12], requiring just n scalars to represent a matrix of n^2 elements. Generalizing along these lines, periodically sampled individual rows of C_{ij} might be assumed to hold over a surrounding row range. Dorr [11] has also recently proposed an extremely low-dimensional parameterization of the interdomain communication by means of Lagrange multipliers, in which it is affordable to explicitly construct and directly solve a generalized capacitance matrix. Multicomponent problems would require a "blocking" of these ideas.
4. *Efficient methods for nonunidirectional convection.* The restriction of the MSC to block diagonal form in the interest of clean parallelism may be too steep a price to pay in convergence rate for problems in which placing cuts through convective recirculation zones cannot be avoided. One is therefore left with the necessity of introducing global coupling at a lower level. Implementation issues related to the evaluation of and solution of the MSC systems with off-diagonal blocks have yet to be addressed.

It would also be of interest, considering the cost and complexity of constructing hierarchical MSC-based preconditioners, to ask how well three other classes of preconditioners do on the same problems: (1) non-hierarchical coupled preconditioners, (2) non-hierarchical decoupled preconditioners with overlap, and (3) non-hierarchical decoupled preconditioners without overlap. An example of the first would be GMR/ILU in which the incomplete factorization is carried out over the entire domain, rather than within subdomains. Since no interfacial blocks requiring approximation are introduced, this technique, appropriately blocked at the gridpoint level, will have a better iteration count than MSC-based techniques on most problems; however, its parallel efficiency can be so low in some problems that it is inferior even to unpreconditioned GMR [3] as a parallel algorithm. An example of the second Schwarz-type method would be GMR/ILU in which the incomplete factorization is carried out independently over overlapping subdomains, followed by some arbitration scheme for the common degrees of freedom. The disadvantages of such methods are the redundant effort expended on the common unknowns and the trade-off (in the parallel context) between sequential bottlenecks and the use of "lagged" data in the overlap regions. An example of the third method would be the use of

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} \tilde{A}_{11} & 0 \\ 0 & \tilde{A}_{22} \end{pmatrix} \tag{14}$$

where the $\tilde{A}_{ii}$ are ILU approximations, instead of (1) and (2). In this method, the creation of independent threads of computation in the preconditioner comes at the expense of severing the interdomain coupling altogether. It therefore parallelizes with virtually no overhead, but can suffer in iteration count as the granularity of the decomposition is refined and more and more of the coupling is discarded, the limit being block-point diagonal preconditioning. Some experiments with this technique have been reported in [1] and [2], and we have also tested it on all of the problems reported above, since it involves only minor modifications to the MSC-based code. For a given n and p, it is nearly always inferior in iteration count, but not always so in CPU time, since the cost of constructing and applying the preconditioning is less. Guidelines on when the interface coupling has sufficient incremental value to warrant (1)-(2) over (14) are lacking except in obvious special cases.

There is no consideration of complex domain geometry in the present work although ease of generalization to this case is an equally relevant motivation. The case of problems requiring too much memory to be managed by just one processor is another one in which the proposed technique is attractive, despite possible inefficiency relative to a global technique.

References.

[1] C. Ashcraft and R. Grimes, *On Vectorizing Incomplete Factorizations and SSOR Preconditioners,* Technical Report ETA-TR-41, Boeing Computer Services, December 1986.

[2] C. Ashcraft, *Domain Decoupled Incomplete Factorizations,* Technical Report ETA-TR-49, Boeing Computer Services, April 1987.

[3] D. J. Baxter, *personal communication,* 1987.

[4] P. E. Bjorstad and O. B. Widlund, *Iterative Methods for the Solution of Elliptic Problems on Regions Partitioned into Substructures,* Technical Report 136, Courant Institute of Mathematical Sciences, NYU, September 1984.

[5] J. H. Bramble, J. E. Pasciak and A. H. Schatz, *The Construction of Preconditioners for Elliptic Problems by Substructuring, I,* Math. Comp., 47 (1986), pp. 103–134.

[6] T. F. Chan, *personal communication,* 1987.

[7] T. F. Chan and D. Resasco, *A Survey of Preconditioners for Domain Decomposition,* Technical Report 414, Computer Science Dept., Yale University, September 1985. In Proceedings of the IV Coloquio de Matemáticas del CINVESTAV, Workshop in Numerical Analysis and its applications, Taxco, Mexico, Aug. 18-24, 1985.

[8] P. Concus, G. H. Golub and G. Meurant, *Block Preconditioning for the Conjugate Gradient Method,* Technical Report 14856, Lawrence Berkeley Laboratory, July 1975.

[9] R. W. Cottle, *Manifestations of the Schur Complement,* Lin. Alg. Appl., 8 (1974), pp. 189–211.

[10] A. R. Curtis, M. J. Powell and J. K. Reid, *On the Estimation of Sparse Jacobian Matrices,* J. Inst. Math. Appl., 13 (1974), pp. 117–119.

[11] M. R. Dorr, Domain Decomposition via Lagrange Multipliers, *Second International Symposium on Domain Decomposition Methods,* 1988.

[12] G. H. Golub and D. Mayers, *The Use of Pre-Conditioning over Irregular Regions,* 1983. Lecture at Sixth Int. Conf. on Computing Methods in Applied Sciences and Engineering, Versailles, Dec. 1983.

[13] W. D. Gropp and D. E. Keyes, *Complexity of Parallel Implementation of Domain Decomposition Techniques for Elliptic Partial Differential Equations,* SIAM J. Sci. Stat. Comp., 9 (1988), pp. 312-326.

[14] D. E. Keyes and W. D. Gropp, *A Comparison of Domain Decomposition Techniques for Elliptic Partial Differential Equations and their Parallel Implementation,* SIAM J. Sci. Stat. Comp., 8 (1987), pp. s166-202.

[15] D. E. Keyes and M. D. Smooke, *Flame Sheet Starting Estimates for Counterflow Diffusion Flame Problems,* J. Comp. Phys., 73 (1987), pp. 267–288.

[16] ————, A Parallelized Elliptic Solver for Reacting Flows, A. K. Noor ed., *Parallel Computations and Their Impact on Mechanics,* ASME, 1987, pp. 375–402.

[17] T. A. Manteuffel, *The Tchebychev Iteration for Nonsymmetric Linear Systems,* Numer. Math., 28 (1977), pp. 307–327.

[18] J. A. Meierink and H. A. Van der Vorst, *Guidelines for the Usage of Incomplete Decompositions in Solving Sets of Linear Equations as they Occur in Practical Problems,* J. Comp. Phys., 44 (1981), pp. 134–155.

[19] J. S. Przemieniecki, *Matrix Structural Analysis of Substructures,* AIAA J., 1 (1963), pp. 138–147.

[20] Y. Saad and M. Schultz, *Parallel Implementation of Preconditioned Conjugate Gradient Methods,* Technical Report YALEU/DCS/RR–425, Computer Science Dept., Yale University, October 1985.

[21] ————, *GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems,* SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856–869.

[22] M. D. Smooke, *Solution of Burner-Stabilized Pre-Mixed Laminar Flames by Boundary Value Methods,* J. Comp. Phys., 48 (1982), pp. 72–105.

[23] R. R. Underwood, *An Approximate Factorization Procedure Based on the Block Cholesky Decomposition and its Use with the Conjugate Gradient Method,* Technical Report NEDO-11386, General Electric Co., Nuclear Energy Div., 1976.

[24] H. F. Walker, *Implementation of the GMRES Method Using Householder Transformations,* SIAM J. Sci. Stat. Comp., 9 (1988), pp. 152–163.

[25] J. W. Watts, III, *A Conjugate Gradient-Truncated Direct Method for the Iterative Solution of the Reservoir Simulation Pressure Equation,* Soc. Petrol. Engin. J., 21 (1981), pp. 345–353.

[26] O. B. Widlund, Iterative Substructuring Methods: Algorithms and Theory for Elliptic Problems in the Plane, R. Glowinski, G. H. Golub, G. A. Meurant and J. Periaux ed., *First International Symposium on Domain Decomposition Methods for Partial Differential Equations,* SIAM, 1988, pp. 113–128.