Parallel multi-level solvers for spectral element methods

P. F. Fischer [*]

Abstract

Efficient solution of the Navier-Stokes equations in complex domains is dependent upon the availability of fast solvers for sparse linear systems. For unsteady incompressible flows, the pressure operator is the leading contributor to stiffness, as the characteristic propagation speed is infinite. In the context of operator splitting formulations, it is the pressure solve which is the most computationally challenging, despite its elliptic origins. We seek to improve existing spectral element iterative methods for the pressure solve in order to overcome the slow convergence frequently observed in the presence of highly refined grids or high-aspect ratio elements. The new algorithm incorporates an enriched coarse-grid operator which admits anisotropic resolution within elements, and a new parallel solution technique which mitigates the additional overhead of the enlarged coarse-grid system.

Key words: spectral element methods, domain decomposition, sparse matrices, parallel algorithms.

AMS subject classifications: 65M70,65Y05,65M55.

1 Introduction

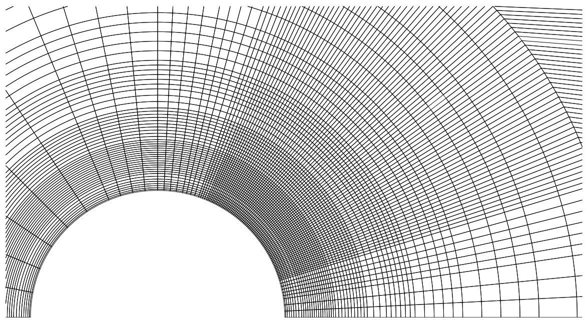
We consider the problems encountered in large-scale spectral element simulations of unsteady incompressible flows. Accurate simulation of even two-dimensional flows can require hundreds of thousands of grid points when the Reynolds number is on the order of 10^4. In the spectral element method, this elevated resolution can be attained by either increasing K, the number of elements, or increasing N, the order of approximation within each element. In practice, it is common to keep the order at a moderate level, i.e., in the range $N = 5$ to 15, and increase the number of elements to capture increasing physical and geometrical complexity.

We have followed this approach in a number of recent high-Reynolds number simulations of start-up flow past a cylinder. Fig. 1a shows an example of the mesh used to compute the early evolution of wake vortices at $Re_D = U_\infty D/\nu = 9500$. Fig. 1b shows the vortex structure at a non-dimensional time of $\tau = tU_\infty/D = 3.1$. The drag history, shown in (c), agrees well with the results of [7] which were based upon an adaptive vortex method using up to 10^6 elements. The present calculation used a total of $K = 6112$ spectral elements, with the order varying from $N = 4$ at early times to $N = 9$ at later times.

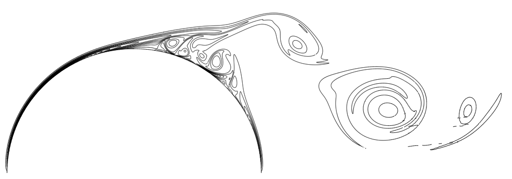
At elevated resolutions, the linear system which imposes the pressure/divergence-free constraint at each time step can become very ill-conditioned and require hundreds to thousands of iterations to reach convergence. For unsteady problems, this computational burden can typically be halved by the projection techniques described in [3]. However further reductions must come through improvements to the iterative solver. We focus here upon development of the two-level deflation based iteration scheme which was proposed by Rønquist in [16]. For highly refined or high-aspect ratio meshes, the convergence rate of this preconditioned/deflated conjugate gradient scheme can be improved by enriching the coarse-grid space to incorporate piecewise (discontinuous) linear or higher-order bases, albeit at a increase in the cost of the repeated coarse-grid solves. We show that the coarse-grid solve cost can be mitigated by a new parallel algorithm for repeated solution of sparse linear systems.

The outline of the paper is as follows. In Section 2, we review the derivation of the system governing the pressure. In Section 3, we reconsider the deflation-based iterative scheme used to solve the pressure system, and show the advantages of an enriched coarse-grid operator. In Section 4, we present a parallel algorithm for the coarse-grid solve with achieves the minimum possible communication complexity and has a computational complexity comparable to standard LU factorizations.

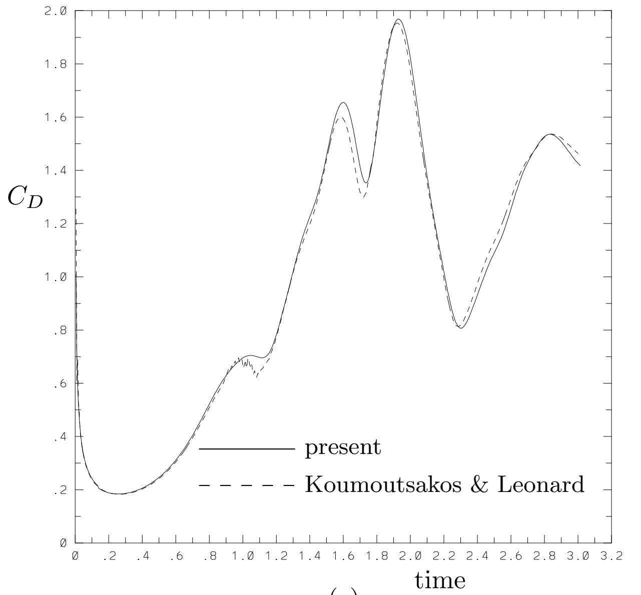
[*]Present address: Division of Applied Mathematics, Brown University, Providence, RI 02912, USA. E-mail: *pff@cfm.brown.edu*



(a)



(b)



(c)

Figure 1: (a) close-up of $K = 6112$ spectral element mesh for computation of start-up flow past a cylinder at $Re = 9500$ (b) vorticity contours at a convective time of $t = 3.10$ (c) drag coefficient C_D vs. non-dimensional time (Computation by G.W. Kruse).

2 Navier-Stokes Implementation

We consider spectral-element solution of the incompressible Navier-Stokes equations:

$$
(1) \qquad \begin{aligned}
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} &= -\nabla p + \tfrac{1}{Re}\nabla^2\mathbf{u} && \text{in } \Omega,\\
\nabla\cdot\mathbf{u} &= 0 && \text{in } \Omega,
\end{aligned}
$$

where $\mathbf{u}$ is the velocity vector, p the pressure, and $Re = \frac{UL}{\nu}$ the Reynolds number based on a characteristic velocity, length scale, and kinematic viscosity.

Spatial discretization is based upon decomposition of the computational domain into K spectral elements which are locally mapped to $[-1,1]^d$ in $\mathbb{R}^d$. Within each element, the geometry, solution, and data are expanded in terms of high-order tensor-product polynomial bases in each coordinate direction. Variational projection operators are used to discretize the elliptic equations arising from a semi-implicit treatment of (1) and a consistent variational formulation is used for the pressure/divergence treatment. The velocity is represented by Nth-order Lagrange polynomials on the Gauss-Lobatto-Legendre quadrature points, with C^0 continuity enforced at element interfaces. The pressure is represented by polynomials of degree $N-2$ based upon the Gauss-Legendre quadrature points; inter-element continuity of pressure is not enforced. Temporal discretization is based upon an operator splitting in which the nonlinear convective terms are treated explicitly via a characteristic/sub-cycling scheme, and the viscous and divergence operators are treated implicitly. The discretization leads to the following linear Stokes problem to be solved at each time step:

$$
(2) \qquad \begin{aligned}
H\,\underline{u}_i - D_i^T\,\underline{p} &= B\,\underline{f}_i, \qquad i = 1,...,d \quad ,\\
D_i\,\underline{u}_i &= 0 \quad .
\end{aligned}
$$

Here, H is the discrete equivalent of the Helmholtz operator, $\{-\frac{1}{Re}\nabla^2 + \frac{1}{\Delta t}\}$; B is the mass matrix associated with the velocity mesh; $\mathbf{D} = (D_1,...,D_d)$ is the discrete gradient operator; and underscore refers to basis coefficients. Further details of spectral element discretizations for the Navier-Stokes equations may be found in [8].

The solution of (2) is simplified by a Stokes operator splitting which decouples the viscous and pressure/divergence constraint [9]. This splitting leads to the solution of a standard Helmholtz equation for each velocity component, while the resulting system for the pressure is similar to (2) save that H is replaced by $\frac{1}{\Delta t}B$. The resulting system can be efficiently treated by formally carrying out block Gaussian elimination (Uzawa decoupling) for $\underline{p}$, leading to:

$$
(3) \qquad E\underline{p} = \underline{g}\,,
$$

where

$$
(4) \qquad E = -\sum_{i=1}^{d} D_i\,B^{-1}D_i^T\,,
$$

and $\underline{g}$ is the inhomogeneity resulting from the time-split treatment of (1). E corresponds to a consistent Poisson operator for the pressure and, though symmetric-positive definite, is less well conditioned than the Helmholtz problems for the velocity components. Consequently, solution of (3) dominates the Navier-Stokes solution time. The advantage of the Stokes splitting is that no system solves are required when applying E, as B is diagonal.

3 Deflation

The consistent Poisson problem (3) is solved via a two-level iteration scheme developed by Rønquist [16] in which a coarse-grid operator is folded into a global conjugate-gradient iteration through deflation [11, 12]. The coarse (subscript c) and fine (subscript f) decomposition is effected through a subdomain-motivated prolongation operator $J \in \mathbb{R}^{m\times n}$, where $m = K(N-1)^d$ is the number of pressure degrees-of-freedom, and n is the dimension of the coarse-grid approximation space. The column space of the prolongation operator J is intended to approximate the span of the low eigenmodes of the E system. The pressure is then expressed as $\underline{p} = J\underline{p}_c + \underline{p}_f$, leading to an algebraic reformulation of the original problem as solvable fine and coarse subproblems,

$$(5) \qquad E_f \underline{p}_f = \underline{g} - J\, E_c^{-1} J^T \underline{g} \,,$$

$$(6) \qquad E_c \underline{p}_c = J^T \underline{g} - J^T E \underline{p}_f \,,$$

respectively. Here $E_f = E - E\, J\, E_c^{-1} J^T E$, and $E_c = J^T E\, J$. Each application of the fine grid operator requires the solution of the relatively small $(n \times n)$ system, E_c. The fine system (5) is solved by conjugate-gradient iteration restricted to the complement of *span*$\{J\}$, where *span*$\{\}$ denotes the column space of the argument. Once $\underline{p}_f$ is established, the coarse-grid problem is solved (directly) for $\underline{p}_c$, and the procedure is complete. With appropriate application of a local, element-based preconditioner to E_f, the condition number of the fine system is significantly reduced relative to the originating E matrix.

In [16] J was chosen to map $n = K$ element-piecewise-constant functions to the m nodes of the underlying spectral element discretization, with $\underline{p}_c$ therefore representing the average pressure within each element. It was shown that for a one-dimensional spectral element discretization, the resultant fine system has a condition number of unity when coupled with a *local* (block Jacobi) spectral preconditioner, E_J. In addition, it was observed that the iteration count to solve (5) is linearly dependent upon N and independent of K for highly regular two-dimensional discretizations. However, Couzy [2] observed that there is some K dependence of the convergence rate, particularly in conjunction with non-unit aspect ratio elements. We have also observed convergence rate degradation in the presence of extreme refinement or high-aspect ratio elements.

It was suggested in [2, 16] that the source of the difficulty lies with the use of block-Jacobi preconditioning. It is well known (e.g., [17]) that the preconditioner should have some subdomain overlap in order to obtain order-independent convergence and block-Jacobi does not provide this. Unfortunately, the $\mathcal{L}^2$ pressure operator (4) does not readily admit construction of a preconditioner with overlap in the spectral element case. Couzy [2] suggests to improve upon the current formulation by imposing Neumann velocity boundary conditions at the element interfaces when generating the local preconditioner rather than zero-Dirichlet as described in [16]. An alternative, which we pursue here, and which is not exclusive of the Neumann-derived boundary conditions (or overlapping methods), is to increase inter-element coupling by increasing the number of modes which are computed as part of the coarse-grid solution. This of course increases the coarse-grid solve costs, but these can be mitigated with appropriate sparse matrix techniques. We develop a fast parallel direct solver for the coarse problem in the next section.

The coarse-grid space is enriched by enlarging the column space of the prolongation operator, J, to include piecewise linear or higher-order functions within each spectral element. In our variational formulation of the Stokes problem, inter-element continuity of the pressure is not strictly enforced. Therefore, it is admissible to use any element-piecewise discontinuous bases for prolongation, resulting in a block diagonal structure for J:

$$J = \begin{bmatrix} J_1 & & & & \\ & J_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & J_K \end{bmatrix} ,$$

Here, each J_k is an $(N-1)^d \times l$ block associated with element k, and l is the number of modes represented within each element. Presently, we take the basis to be a tensor product of the first $\sqrt[d]{l}$ polynomial modes in each spatial direction. Note that the relaxation of strict continuity implies that we need not have the same coarse-grid approximation within each element. It is also possible to choose an anisotropic basis, e.g., for elements having high-aspect ratios. We have built this flexibility into our code and will investigate its potential the future. Here we only consider a uniformly enriched coarse-grid space.

As noted in [10, 12, 16] effective implementation of deflation requires suppression of spurious modes in the null space of E_f during the conjugate gradient iteration. This is particularly true if the preconditioner does not share the same null space as E_f [10]. In [4], we demonstrated the advantages of replacing E_J with a block preconditioner, E_F, based upon finite element solution of a local Poisson problem within each of the spectral elements. Whereas it is relatively easy to construct a (spectral based) block-Jacobi preconditioner, E_J, sharing the same null space as E_f, the same is not true for E_F, as it is based upon a different discretization. However, the necessary symmetry and null space conditions can be satisfied by defining a new preconditioner, $E_P^{-1} \equiv PE_F^{-1}P^T$, where $P = P^T$ is the orthogonal projection matrix given by

$$P \quad = \quad I_m - J(J^TJ)^{-1}J^T \quad , \tag{7}$$

and I_m is the $m \times m$ identity matrix. Null space control in the preconditioned conjugate gradient algorithm is thereby assured through the application of E_P^{-1} to the residual vector in the preconditioning step. Note that the block structure of J ensures that multiplication of a vector by P is a strictly local operation. The application of P is further simplified if the columns of J are orthonormalized such that $J^TJ = I_n$.

Results for the Enriched Coarse-Grid Space:

We demonstrate the effectiveness of the enriched coarse-grid for two model problems: unsteady Stokes flow in a square box, and start-up flow past a cylinder.

The first problem is identical to the unsteady Stokes flow problem considered in [16], save that the present implementation employs finite element preconditioning as described in [4]. The two-dimensional problem is specified on $\Omega = [-1, 1]^2$ with homogeneous velocity boundary conditions, viscosity $\nu = 0.1$, and a body force $\mathbf{f} = (-0.6y, 0)^T$. The discretization consists of K square elements, each of order $N = 7$, and a time step $\Delta t = 0.1$.

Table 1 shows the number of preconditioned conjugate gradient steps required for the first time step for several values of l and K. The corresponding dimension of the original system, E, and the coarse-grid system, E_c, are also given. The heading E indicates the number of iterations required to reduce the initial residual of (3) by 10^5; the heading E_f indicates the number of iterations required to reduce the initial residual of (5) by the same magnitude. Note that, while order independent convergence is obtained when for the E_f system when $l = 1$, the same is not true for the original E system. Although the norm of the initial residual, $(\underline{g}^TB^{-1}\underline{g})^{\frac{1}{2}}$, is the same for all cases, the norm for the right-hand side of (5) actually increases with K for the case $l = 1$.

Table 1: Iteration count for unsteady Stokes flow

l	K	E_f	E	dim$\{E\}$	dim$\{E_c\}$
1	4	20	20	144	4
1	16	30	32	576	16
1	64	32	37	2304	64
1	256	32	40	9216	256
1	1024	32	42	36864	1024
4	4	18	18	144	16
4	16	26	26	576	64
4	64	28	29	2304	256
4	256	28	30	9216	1024
4	1024	27	29	36864	4096
9	4	15	15	144	36
9	16	22	22	576	144
9	64	25	25	2304	576
9	256	26	24	9216	2304
9	1024	26	22	36864	9216

We next consider solution of the Navier-Stokes equations for flow past a cylinder in the half-domain $\Omega = [-10, 28] \times [0, 15]$. A cylinder of diameter $D = 1$ is centered at the origin. The Reynolds number is $Re = DU/\nu = 80$, where $(U, 0)$ is the free-stream velocity taken as the initial condition and inflow boundary condition at $x = -10$. Symmetry boundary conditions are imposed at $y = 0$ and $y = 15$ with Neumann-velocity (outflow) boundary conditions at $x = 28$. The free stream velocity is $U = 1$ and the time step is $\Delta t = .025$. The base mesh ($K = 93$) is shown in Fig. 2. The $K = 372$ and $K = 1488$ meshes are obtained through successive quarterings of the elements in the base configuration.

Table 2 shows the number of iterations required to reduce the residual of the E_f and E systems by five orders of magnitude for the first time step. Again, the norm of the initial residual of the E system (3) is independent of K. In this case, K-independence convergence rates are not obtained for any choice of l, though the reduction in iteration

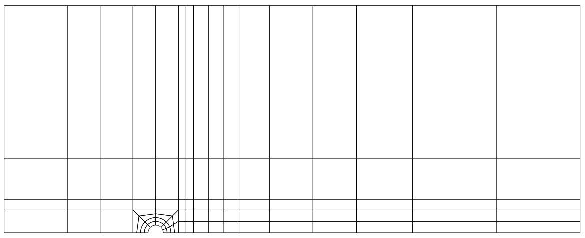


Figure 2: Spectral element mesh ($K = 93$) for Problem 2

Table 2: Iteration count for cylinder flow

l	K	E_f	E	$\dim\{E\}$	$\dim\{E_c\}$
1	93	115	107	3348	93
1	372	185	184	13392	372
1	1488	265	279	53568	1488
4	93	82	69	3348	372
4	372	116	102	13392	1488
4	1488	150	138	53568	5952
9	93	63	51	3348	837
9	372	84	73	13392	3348
9	1488	108	89	53568	13392

count for the $K = 1488$, $l = 4$, case is clearly significant.

4 Parallel coarse-grid solve

We focus here on fast solvers for the coarse-grid problem. In practice, the coarse-grid system derived from piecewise constant prolongation operators is sufficiently small so that direct inversion of the system leads to an efficient parallel solution strategy based upon matrix-vector multiplication (e.g., [4]). However, the enriched coarse-grid space outlined above leads to significantly larger system and direct inversion is no longer viable. To circumvent this, we have developed a new solution approach which retains the parallelism of matrix-vector products, yet avoids the $O(n^2)$ costs of inversion. In the following, our primary emphasis will be on system *solution* rather than factorization, as the latter is amortized over many time steps and iterations. For notational convenience, we take the coarse grid operator to be the $n \times n$ symmetric positive definite matrix denoted by A, and consider the problem of solving $A\underline{x} = \underline{b}$ on a p-processor distributed memory computer using a single program, multiple data (SPMD) programming model with message-passing.

To understand the relevant software design parameters, it is important to recognize that the coarse-grid problem is relatively fine-grained, i.e., the dimension of the problem, n, often scales as a small multiple of p. Interprocessor messages are therefore typically small, implying that startup (or latency) costs may dominate the communication times. Consequently, a potentially important design metric is the required *number* of messages, rather than the operation count or total amount of inter-processor data traffic (bandwidth). If the contention-free time to communicate a message of length w words is given by $t_c(w) = \alpha + \beta w$, we can characterize a message as "short" whenever $w \leq w_2$, where $w_2 \equiv \alpha/\beta$ is the message length corresponding to a transmission time of twice the lowest possible value. Typically, $w_2 \simeq 100 - 500$, for 32-bit words.

It is important to note that the elliptic nature of the originating problem implies that each component of the right-hand side will have a non-trivial impact on the solution. If the right-hand side vector is distributed, a lower bound on the number of message cycles to invoke each solve is therefore $\log_2 p$. Algorithms achieving this bound can be considered optimal in the fine-grained, short message limit.

For the development of the parallel solution scheme, we assume that vectors and matrix columns are distributed in the same fashion as spectral elements, i.e., for each entry in a column vector, there is a corresponding spectral element. We further assume that A is sparse; each non-zero in a given row is associated with an element adjacent to the element corresponding to that row. This holds only for the piecewise constant prolongation operator ($l = 1$, $n = K$) of the previous section. The strategy for higher-order coarse grid operators ($l > 1$) is the same – each degree-of-freedom in the following discussion is simply replaced by a clique of l fully coupled degrees-of-freedom.

It is well known (e.g., [6]) that parallel solution of the coarse grid problem is hampered by the inherent sequentiality of the forward and backward substitution phases of standard LU (or LL^T) factorizations. If n (and consequently, p) is sufficiently small, it is feasible to store, factor, and solve the system redundantly on each processor. This often practiced approach has the advantage that it is easy to implement and requires a per-solve communication time of only $(\alpha \log_2 p + \beta n)$ to gather a copy of $\underline{b}$ onto each processor. If the local direct solution strategy is based upon standard banded solvers, the computational complexity is $4ns$ operations per solve, for a matrix of bandwidth s.

For large numbers of processors and relatively small systems (e.g., $p > 128$, $n < 5000$), we have shown that computing the full inverse of A can be far more effective than solving the system redundantly [4]. By distributing each column of A^{-1} in the same manner as $\underline{x}$ and $\underline{b}$, the solution can be computed as a parallel matrix-vector product,

$$\underline{x} = A^{-1}\underline{b} \quad , \tag{8}$$

once $\underline{b}$ has been gathered onto each processor. The communication complexity is identical to the previous case. However, the complexity for the computation of the inner-products of the rows of A^{-1} with $\underline{b}$ is now $2n^2/p$; parallelism has been introduced to this phase of the solution. Whenever $p > n/2s$, the full inverse approach will be superior to redundant banded factorization.

Unfortunately, memory limitations restrict the above techniques to matrices of order n less than a few thousand. Ideally, one would like a solution scheme which retains the low $O(\log p)$ communication cost of the previous approaches, has the full parallelism of matrix-vector products, yet has the sparsity of LU factorization. The following construction attains all of these goals.

Let $X = (\underline{x}_1\ \underline{x}_2\ \dots\ \underline{x}_n)$ be a square matrix with columns satisfying

$$(9) \qquad \underline{x}_i^T A \underline{x}_j \quad = \quad \delta_{ij} \quad ,$$

where δ_{ij} is the Kronecker delta. Then $XX^TA\underline{x} = XX^T\underline{b}$ is the A-orthogonal projection of $\underline{x}$ onto $\mathbb{R}^n$, and XX^T constitutes a factorization of A^{-1}. If each $\underline{x}_i$ is distributed in the same manner as $\underline{b}$, the two-step computation of

$$(10) \qquad \underline{x} = XX^T\underline{b}$$

has the desired communication complexity. In the first step, the vector $\underline{c} = X^T\underline{b}$ is computed by locally forming n partial inner-products on each processor, followed by a $\log_2 p$ gather-scatter of an n-vector to sum and redistribute the entries of $\underline{c}$. Assuming that X is a full matrix, the computational complexity for this step is $2n^2/p$. If an interleaved sum-and-redistribute operation is used, the communication complexity is $(\alpha \log_2 p + \beta n \log_2 p)$ to generate an *entire* copy of $\underline{c}$ on each processor. In the second step, the computation of $\underline{x} = X\underline{c}$ requires $2n^2/p$ operations and no communication.

The overall complexity can be significantly reduced if we can choose a (quasi-) sparse basis for the columns of X. By quasi-sparse, we imply that X will have $O(n^\gamma)$ entries, with $1 < \gamma < 2$, as opposed to strictly $O(n)$ entries as is generally associated with the definition of a sparse matrix. We will nonetheless refer to X as being simply "sparse." The following geometric arguments show that it is always possible to generate a sparse factor X if A is sparse.

Let $\hat{\underline{e}}_i$ and $\hat{\underline{e}}_j$ be the ith and jth column of of the identity matrix, I_n, and $\mathcal{N}_i$, the *neighborhood* of i, be the set of indices corresponding to columns with non-zero entries in row i of A. Then,

$$(11) \qquad \hat{\underline{e}}_j^T A \hat{\underline{e}}_i \quad = \quad 0 \quad \forall\, j \notin \mathcal{N}_i \quad .$$

Geometrically, this corresponds to the situation shown in Fig. 3a. It is clear that at least $n/\max(\#\mathcal{N}_i)$ of the unit vectors are A-conjugate to one another, where $\#\mathcal{N}_i$ denotes the cardinality of $\mathcal{N}_i$. The generation of a sparse basis for X starts with finding a maximal (or near-maximal) set of k_1 A-conjugate unit vectors and normalizing them appropriately. The first k_1 columns of X will have *only one non-zero entry.*

Additional entries in X are generated by Gram-Schmidt orthogonalization. Let $X_k = (X_{k-1}\, \underline{x}_k)$ denote the $n \times k$ matrix with columns $(\underline{x}_1\, \underline{x}_2 \dots \underline{x}_k)$, and let $V = (\underline{v}_1\, \underline{v}_2 \dots \underline{v}_n)$ be an appropriate permutation of the identity matrix. Then the procedure:

```
for  $i = 1, \dots, n$:
    $\underline{w} := \underline{v}_i \; - \; X_{i-1}X_{i-1}^T A \underline{v}_i$
    $\underline{x}_i := \underline{w}/||\underline{w}||_A$
    $X_i := (X_{i-1}\ \underline{x}_i)$
end for
```

ensures that X is the desired factor of A^{-1}. For $i \le k_1$ the projection $X_{i-1}X_{i-1}^T A\underline{v}_i$ will be void and $\underline{x}_i$ will simply be a multiple of $\underline{v}_i$. As k increases beyond k_1, X_k will begin to fill in. At the cost of additional communication overhead, a more stable modified Gram-Schmidt procedure may replace the projection step, $\underline{w} := \underline{v}_i - X_{i-1}X_{i-1}^T A\underline{v}_i$, given above.

Following, e.g., [5, 15], an efficient procedure for selecting the permutation matrix, V, can be developed by defining separators which recursively divide the domain (or graph) associated with A into nearly equal subdomains. The first such separator is shown in Fig. 3b. Since the stencil for A does not cross the separator, it is clear that every unit vector $\hat{\underline{e}}_i$ associated with the left half of the

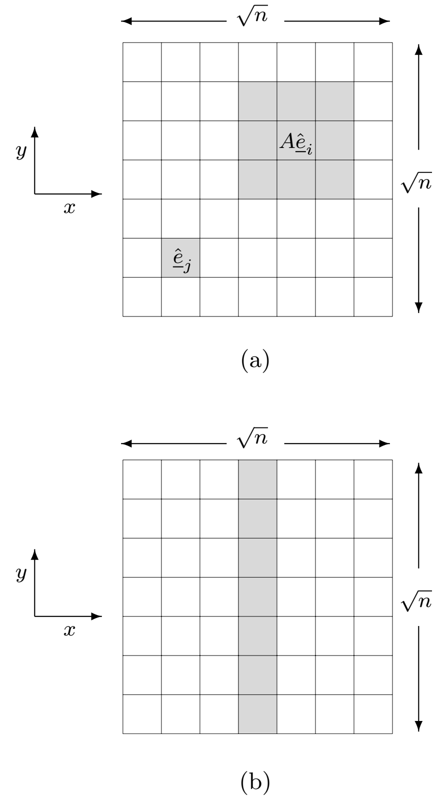
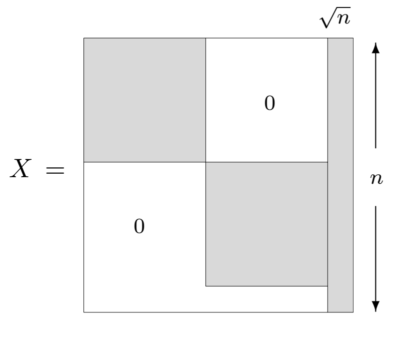
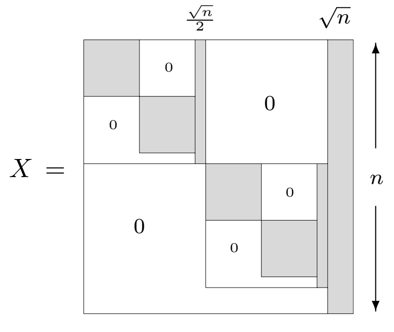


Figure 3: (a) Geometric support (shaded) of orthogonal vectors $\hat{\underline{e}}_j$ and $A\hat{\underline{e}}_i$. (b) Support of spectral element separator set.



(a)



(b)

Figure 4: (a) zero/fill structure for X resulting from ordering suggested by Fig. 3b. (b) zero/fill structure after second round of recursion.

domain in Fig. 3b is A-conjugate to every unit vector $\underline{\hat{e}}_j$ associated with the right half. If V is arranged such that vectors associated with the left half of the domain are ordered first, vectors associated with the right half second, and vectors associated with separator last, then the above Gram-Schmidt procedure will generate a matrix X with worst-case fill depicted by Fig. 4a. (X is shown here with the rows reordered according to the permutation used for the columns of V.) The procedure can be repeated to order the vectors within each subdomain, giving rise to the structure shown in Fig. 4b. Repeating the procedure recursively will lead to a total storage bounded by $3n\sqrt{n}$ for a $\sqrt{n} \times \sqrt{n}$ array of spectral elements.

From (10), it is clear that the computational complexity of each solve is proportional to the amount of fill in the factor X. Depending on the spectral element-to-processor distribution, we can expect a computational complexity of $O(n^{\frac{3}{2}}/p)$ for the domain shown in Fig. 3b – a clear gain over the $O(n^2/p)$ cost incurred by the matrix inverse approach (8). Moreover, the communication cost can be shown to be only $(\alpha \log_2 p + O(\sqrt{n})\beta \log_2 p)$. Similar arguments in three-dimensions lead to computational and communication complexities of $O(n^{\frac{5}{3}}/p)$ and $(\alpha \log_2 p + O(n^{\frac{2}{3}})\beta \log_2 p)$, respectively. Note that for a typical value of $\alpha/\beta \simeq 100$, communication bandwidth does not play a role in the two-dimensional case until $n \simeq 10^4$; the communication cost in the solution of smaller systems is dominated by latency.

For more complex two- or three-dimensional meshes, separator sets can be found with standard graph-splitting techniques, e.g., recursive coordinate bisection, or one of the many variants of recursive spectral bisection (RSB). In this application, the cost of finding a good separator is amortized over numerous solves and the per-solve cost is a super-linear function of the size of the separators. Consequently, high-quality bisection algorithms are of interest.

We have implemented a parallel version of RSB [15] to order the separator sets and, ultimately, the columns of V. In RSB, the initial partition is derived from the second lowest eigenmode (the Fiedler vector) of the graph-Laplacian of A. The graph-Laplacian is defined as the matrix $\mathcal{L}$ which has the same sparsity pattern as A, with each off-diagonal entry in $\mathcal{L}$ having value -1 and each diagonal entry equal to the number of off-diagonal entries. The domain is partitioned according to values above and below the median entry in the Fiedler vector. Once this partition is established, the procedure is repeated on each subdomain.

To effect a tripartite decomposition of the domain into a separator and two disjoint subdomains, it is useful to base RSB on the dual of the coarse-grid operator, i.e., the graph-Laplacian of the spectral element *vertices* rather than the element centroids. A spectral element is then deemed to be in the separator set if the Fiedler vector entries corresponding to the element's vertices straddle the median.

The graph-Laplacian satisfies a zero row-sum condition and is symmetric semi-positive definite. Computation of the low eigenmodes of $\mathcal{L}$ is accomplished via standard Lanczos/conjugate-gradients techniques (e.g., [13]), and hence requires only sparse matrix-vector product software (e.g., [14]). After each separator set is found, $\mathcal{L}$ can be updated by zeroing matrix entries which correspond to deleted entries in the original graph, and adjusting the diagonal entries accordingly. Consequently, the parallel gather-scatter index sets required for matrix-vector products need not be changed with each round of recursion. We typically recur until the minimum number of elements in a subdomain is equal to four, and then use a greedy algorithm to color the remaining entries.

The approach outlined above for generation of sparse factorizations of A^{-1} is strongly linked to nested-dissection factorizations of sparse matrices [5] and to the minimal factor algorithms suggested by [1]. It can readily be shown that there exists a row permutation matrix R such that RX is lower triangular. Then, $(RX)(X^TR^T)$ is the Cholesky factorization of $RA^{-1}R^T$. Strikingly, this implies that while A^{-1} is full, there exists a symmetric permutation of A^{-1} for which its Cholesky factorization is (quasi-) sparse, assuming that A was sparse initially. A trivial example of this can be seen for a symmetric-positive definite tridiagonal matrix, T. The Cholesky factorization of $T = LL^T$ is sparse, but the inverse of L is full lower-triangular. However, if one first permutes T with a nested dissection ordering, the Cholesky factor of the permuted matrix will have an inverse having only $n \log_2 n$ non-zeros.

Results for the Sparse Factorization:

We have implemented the RSB routine in parallel and used it to generate separator sets for several two- and three-dimensional meshes. A close-up of a separator set generated after four rounds of RSB is shown in Fig. 5. With piecewise-constant prolongation ($l = 1$), the coarse grid dimension for this problem is $n = K = 1692$. The number of non-zeros in X is 118921, which is of the same order as the predicted bound of $3n^{\frac{3}{2}} = 208796$. For larger values of l, the storage and per-solve operation count scales as l^2, while the communication bandwidth cost scales as l. It is clear that as l increases, less compute-intensive algorithms such as Choleski factorizations based on nested-dissection orderings (e.g., [1]) will be of interest, since the operation count will begin to dominate the communication costs. However, we have found that even the full n^2 algorithm, (8), is of interest for n on the order of a few thousand and p on the order of a few hundred. We therefore expect the $n^{\frac{3}{2}}$ algorithm presented here to provide significant gains for larger n.

Preliminary results show the merits and potential limitations of the XX^T factorization in conjunction with the enriched coarse grid operator. We compute start-up flow past a cylinder at $Re = 5000$ using the mesh depicted in Fig. 5. The cylinder diameter and free-stream velocity are both unity, and the time step is $\Delta t = 0.001$. Fig. 6a shows the number of iterations required for the E_f system

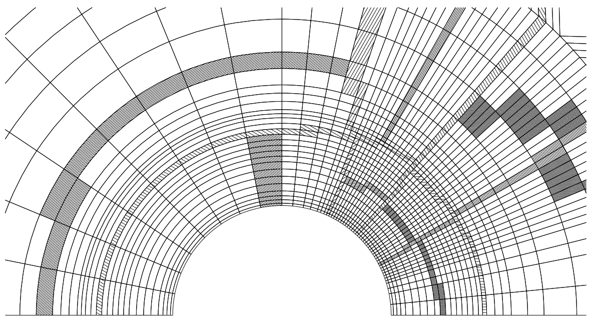
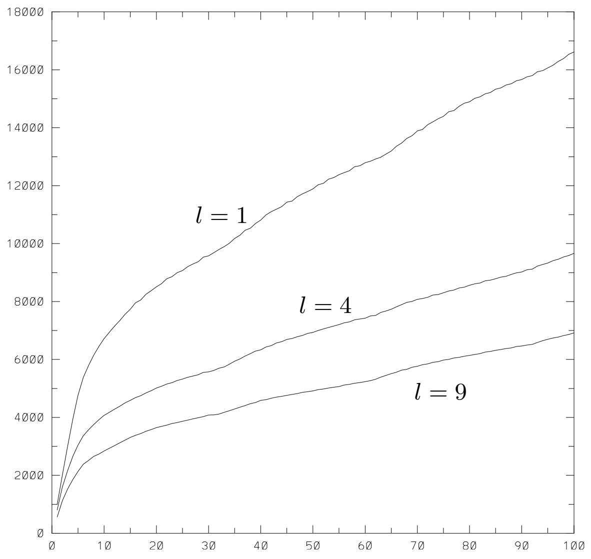
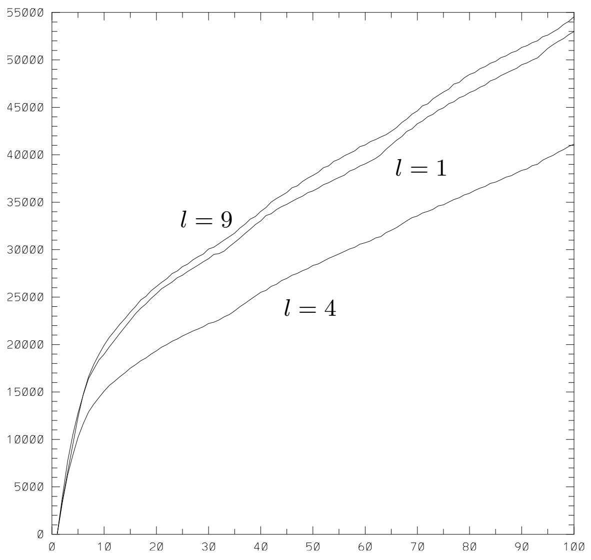


Figure 5: Close up of $K = 1692$ element mesh showing separators (shaded) generated by the first four rounds of RSB. Each quadrilateral is a spectral element.



(a)



(b)

Figure 6: Cumulative pressure iteration count (a) and CPU time in seconds (b) vs. step number for the first 100 time steps of a cylinder start-up calculation on a single processor SGI-Onyx. The Reynolds number is $Re_D = 5000$; the mesh employed is that of Fig. 5, with $K = 1692$, $N = 7$, $l = 1, 4, 9$.

(5) for piecewise constant, linear, and quadratic coarse grid spaces. While there is a clear reduction as l increases, the CPU time on a single-processor SGI Onyx decreases only up to $l = 4$ for this problem, as can be seen in Fig. 6b.

5 Conclusions

We have developed a flexible deflation based iterative algorithm to compute the pressure for large two- and three-dimensional spectral element discretizations of the incompressible Navier-Stokes equations. The method provides significant reduction in iteration count for certain classes of problems, and admits the possibility of tailoring the portion of the solution to be computed directly in order to improve the overall efficiency of the solver. A critical component to the solution of this problem, and many other similar multi-level solvers, is a fast direct method for solving systems of equations in parallel. The approach developed here achieves the minimum possible time complexity in the message-latency dominated limit.

Acknowledgements

This work was supported by the NSF under Grant ASC-9405403 and by the AFOSR under Grant F49620-95-1-0074. Computer time was provided on the Intel Delta by the Center for Research on Parallel Computation at Caltech and on the Intel Paragon at Wright Patterson Air Force Base by the AFOSR. The author would like to thank Gerald Kruse and Henry Tufo for assistance in this work.

References

[1] F. Alvarado, A. Pothen, and R. Schreiber, "Highly parallel sparse triangular solution" *Univ. Waterloo Research Rep. CS-92-51* Waterloo, Ontario (1992).

[2] W. Couzy, "Spectral element discretization of the unsteady Navier-Stokes equations and its iterative solution on parallel computers," Thèse No. 1380, École Polytechnique Fédérale de Lausanne (1995).

[3] P.F. Fischer, "Projection techniques for iterative solution of $A\underline{x} = \underline{b}$ with successive right-hand sides." *ICASE Report No. 93-90*, NASA CR-191571.

[4] P.F. Fischer and E.M. Rønquist, "Spectral Element Methods for Large Scale Parallel Navier-Stokes Calculations" *Comp. Meth. Appl. Mech. Engr.*, 1994.

[5] J.A. George, *Nested dissection of a regular finite element mesh,* SIAM J. Numer. Anal., **15** (1978) pp. 1053-1069.

[6] W.D. Gropp, *Parallel Computing and Domain Decomposition* in Fifth Conference on Domain Decomposition Methods for Partial Differential Equations T.F. Chan, D.E. Keyes, G.A. Meurant, J.S. Scroggs, and R.G. Voigt, eds., SIAM, Philadelphia, PA, 1992.

[7] P. Koumoutsakos and A. Leonard "High-resolution simulations of the flow around an impulsively started cylinder using vortex methods" *J. Fluid Mech.* **296** 10 (1995) pp. 1-38.

[8] Y. Maday, and A.T. Patera, *Spectral element methods for the Navier-Stokes equations* in State of the Art Surveys in Computational Mechanics, A.K. Noor, ed., ASME, New York, 1989.

[9] Y. Maday, A.T. Patera, and E.M. Rønquist, *An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow.* J. Sci. Comput., **5**(4), (1990) pp. 310-37.

[10] Mandel, J "Hierarchical preconditioning and partial orthogonalization for the p-version finite element method", in *3rd Int. Symp. on Domain Decomposition Methods*, T.F. Chan, R. Glowinski, J. Periaux and O .B. Widlund, eds., SIAM, pp. 141-156 (1989).

[11] L. Mansfield, *On the use of deflation to improve the convergence of conjugate gradient iteration* Comm. in Appl. Numer. Meth., **4**, (1988) pp. 151-56.

[12] R.A. Nicolaides, *Deflation of conjugate gradients with applications to boundary value problems* SIAM J. Numer. Anal., **24**(2), (1987) pp. 355-65.

[13] D.P. O'Leary and O. Widlund, "Capacitance matrix methods for the Helmholtz equation on general three-dimensional regions" *Math. Comp.* 33(147), 849-879 (1979).

[14] S. Pissanetzky, *Sparse Matrix Technology* Academic Press, Orlando Florida, 1984.

[15] A. Pothen, H.D. Simon, and K.P. Liou *Partitioning sparse matrices with eigenvectors of graphs* SIAM J. Matrix. Anal. Appl., **11**(3), (1990) pp. 430-52.

[16] E.M. Rønquist, *A Domain Decomposition Method for Elliptic Boundary Value Problems: Application to Unsteady Incompressible Fluid Flow,* in Fifth Conference on Domain Decomposition Methods for Partial Differential Equations T.F. Chan, D.E. Keyes, G.A. Meurant, J.S. Scroggs, and R.G. Voigt, eds., SIAM, Philadelphia, PA, 1992.

[17] M. Dryja and O. Widlund, "Towards a Unified Theory of Domain Decomposition Algorithms for Elliptic Problems," in *Proceedings of the Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, R. Glowinski, J. Periaux, and O. Widlund, eds. (SIAM, Philadelphia, 1990), p. 3.