MATRIX-FREE PRECONDITIONER FOR THE STEADY ADVECTION-DIFFUSION EQUATION WITH SPECTRAL ELEMENT DISCRETIZATION*

HOWARD ELMAN† AND P. AARON LOTT‡

Abstract. We introduce a preconditioning technique based on Domain Decomposition and the Fast Diagonalization Method which can be applied to tensor product based discretizations of the steady advection-diffusion equation. The method is based on iterative substructuring with fast diagonalization to eliminate the interior degrees of freedom. We demonstrate the effectiveness of this preconditioner in numerical simulations using a spectral element discretization.

1. Introduction. The interplay between inertial and viscous forces in a fluid dictates the length scale where energy is transferred, thus determining the resolution required to capture flow information accurately. This resolution requirement poses great theoretical, experimental and computational challenges as the advective nature of the flow begins to dominate diffusive effects. In such flows, advection and diffusion occur on disparate scales, which has motivated the development and use of splitting schemes [1]. The standard method for performing steady and unsteady flow simulations with spectral elements is operator integration factor splitting (OIFS)[5]. They typically involve time integration, even in steady flow simulations. These methods treat advection and diffusion separately; advection components are tackled explicitly using a sequence of small time steps that satisfy the CFL condition, and diffusive components are treated implicitly with larger time steps via a backward differencing formula that couples the system. Such schemes have been successfully applied in a variety of settings including massively parallel turbulence simulations which require long time integration to obtain useful flow statistics.

This paper discusses a new approach for simulating steady flows by treating the advective and diffusive components together. Our method builds on ideas from iterative substructuring by exploiting fast diagonalization to eliminate degrees of freedom in elemental interiors. The efficiency of this method is centered on two advances: the first is the use of an accurate element-based high order matrix-free discretization to construct accurate discrete solutions while minimizing memory requirements; the second is the use of fast iterative solvers via effective preconditioners that take into account memory hierarchies and efficient cache use to enable processor performance.

2. Spectral Element Discretization. The spectral element method is a Galerkin method based on the method of weighed residuals, in which a weak form integral equation is solved. When multiple elements are used, the integral equation is broken up into a summation of the integrals on each element. The element-based integrals are then approximated by numerical quadrature. In particular, the solution is represented by a basis of high order Legendre polynomials at Gauss-Legendre-Lobatto quadrature nodes. The resulting system of linear matrix equations numerically represents the original integral equation on each element. Rectangular elements allow each elemental system to be represented via tensor products of one-dimensional operators. Inter-element coupling of these matrix equations ensures continuity

* This work was supported by the U. S. Department of Energy under grant DEFG0204ER25619, and by the U. S. National Science Foundation under grant CCF0726017.

†Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, elman@cs.umd.edu

‡Applied Mathematics and Scientific Computation Program, University of Maryland, College Park, MD 20742, palott@ipst.umd.edu

along elemental boundaries. These inter-element couplings can be enforced by either constructing a fully coupled sparse linear system of equations, or by performing a gather-scatter operation that sums the solution along element boundaries after element-based matrix-vector products are performed. This gather-scatter operation coupled with the tensor product formulation of elemental operators yields a matrix-free discretization, in which only matrices associated with one-dimensional phenomena need to be stored.

Compared to low order methods, spectral methods require about half as many degrees of freedom in each spatial dimension to accurately resolve a flow. The trade-off for this low memory requirement is an increase in computational cost per degree of freedom. By using the Spectral Element Method, we retain the accuracy of spectral methods and gain the flexibility of a matrix-free discretization to invoke cache efficient element-based matrix-matrix calculations [8]. These element-based calculations provide improved parallelism over spectral and low order methods via reduced global communication and small surface to volume ratios on each element. Together, these computational efficiencies offset the added cost per degree of freedom, thus making spectral elements a competitive choice for discretizing the advection-diffusion equation.

3. Steady Advection-Diffusion Equation. The advection diffusion equation governs many flows. This equation can be written as

$$-\nabla^2 u + (\vec{w}\cdot\nabla)u = f \tag{3.1}$$

where u represents velocity, the vector field $\vec{w} = (w_x(x,y), w_y(x,y))^T$ represents advection or wind speed at each point in the domain, and f represents body forces acting on the fluid. Following the discretization strategy discussed in section 2, equation (3.1) is recast in the weak form:

Find $u \in H^1_E$ such that:

$$\int_\Omega \nabla u\cdot\nabla v + \int_\Omega (\vec{w}\cdot\nabla u)v = \int_\Omega fv \qquad \forall v \in H^1_{E_0}. \tag{3.2}$$

To discretize (3.2), we restrict u and v to finite dimensional subspaces by dividing the domain Ω into E non-overlapping subdomains (elements) $\Omega = \cup_{e=1}^{E}\Omega_e$; each subdomain is then discretized using tensor products of Lagrangian interpolants on N degree Legendre polynomials π_N. Thus, on each element the solution is of the form

$$u(x,y) = \sum_{i=1}^{N+1}\sum_{j=1}^{N+1} u_{ij}\pi_{N,i}(x)\pi_{N,j}(y) \tag{3.3}$$

The coefficients u_{ij} correspond to the nodal values of u on the tensored Gauss-Legendre-Lobatto (GLL) quadrature points defining a discrete solution on each element.

The mass matrix is defined to be the block diagonal matrix $M = diag(M_e)$ where M_e is the local mass matrix on an element of size $h_x \times h_y$ is represented via a tensor product of one-dimensional operators

$$M^e = \frac{h_x h_y}{4}\hat{M}\otimes\hat{M}. \tag{3.4}$$

Due to the orthogonality of the basis functions $\pi_{N,i}$ and $\pi_{N,j}$ the mass matrix is diagonal. The one-dimensional operator $\hat{M} = diag(\rho_i) \quad i = 1 : N+1$ where ρ_i is the GLL weight

associated with the i^{th} GLL quadrature node ξ_i. Likewise, the advection-diffusion operator F is represented on each element through this basis, namely

$$F_x^e = (\hat{M} \otimes \frac{h_y}{h_x}\hat{A}) + W_x^e(\hat{M} \otimes \frac{h_y}{2}\hat{D}) \tag{3.5}$$
$$F_y^e = (\frac{h_x}{h_y}\hat{A} \otimes \hat{M}) + W_y^e(\frac{h_x}{2}\hat{D} \otimes \hat{M})$$
$$F^e(w^e) = F_x^e + F_y^e$$

The discrete two-dimensional diffusion operator is formed via tensor products of the one-dimensional second derivative operator $\hat{A}$ with the one-dimensional mass matrix $\hat{M}$. Similarly, the advection operator is formed via a tensor product of the one-dimensional derivative operator $\hat{D}$ with the mass matrix $\hat{M}$, then scaled by the wind speed at each node via the $(N+1)^2 \times (N+1)^2$ diagonal matrices $W_x^e = diag(w_x(\xi_i, \xi_j))$ and $W_y^e = diag(w_y(\xi_i, \xi_j))$. The one-dimensional spectral differentiation matrix $\hat{D}$ is defined as

$$\hat{D}_{ij} = \frac{d\pi_j}{dr} \mid_{r=\xi_i} \qquad i, j \in \{1, .., N+1\}^2, \tag{3.6}$$

and the one-dimensional second derivative operator $\hat{A}$ is defined in terms of the spectral differentiation matrix $\hat{D}$:

$$\hat{A}_{ij} = \sum_{k=1}^{N+1} \hat{D}_{ki}\rho_k\hat{D}_{kj} \qquad i, j \in \{1, .., N+1\}^2. \tag{3.7}$$

With this discretization strategy we obtain the system of linear equations

$$F(\vec{w})u = Mf = b \tag{3.8}$$

where $F(w)$ is the non-symmetric advection-diffusion operator, and M the diagonal mass matrix. This system is solved using an iterative scheme, namely, preconditioned GMRES [6]. In section 5 we discuss the preconditioning methods used to expedite the convergence of GMRES, but first we introduce a fast solver for tensor product based computations.

4. Fast Diagonalization Method (FDM) . The spectral element discretization enables the advection-diffusion equation to be written as sums of tensor products on each element. This form is particularly useful when performing matrix-vector products, and when solving certain elemental systems of equations. The Fast Diagonalization Method (FDM) [4] allows for the solutions of systems in which the coefficient is of order n^d and has a tensor product structure in $O(n^{d+1})$ operations, where d represents the number of spatial dimensions and n represents the number of quadrature points used along each dimension on a single element.

The Fast Diagonalization Method cannot be used with non-constant coefficients as in equation (3.5). However, by constructing a preconditioner for equation (3.8) based on local bi-constant winds, we can use FDM to solve a constant coefficient problem on each element. In the remainder of the section, we demonstrate how the FDM is applied to the resulting equations.

Consider equation (3.5) in the special case where $W_x^e = c_x$ and $W_y^e = c_y$ are both constant on each element. This allows the advection-diffusion operator on each element to be written as

$$F^e(c_x, c_y) = \hat{M} \otimes \hat{F}_x + \hat{F}_y \otimes \hat{M} =: \bar{F^e}. \tag{4.1}$$

We use the fact that $\hat{M}$ is diagonal to apply a transformation to $\bar{F}^e$ that will allow for fast diagonalization. Namely, $\bar{F}^e = \tilde{M}^{1/2}\tilde{F}\tilde{M}^{1/2}$ such that

$$\begin{aligned}\tilde{F} &= \tilde{M}^{-1/2}F\tilde{M}^{-1/2} \\ &= (\hat{M}^{-1/2}\otimes\hat{M}^{-1/2})(\hat{M}\otimes\hat{F}_x+\hat{F}_y\otimes\hat{M})(\hat{M}^{-1/2}\otimes\hat{M}^{-1/2}) \\ &= (I\otimes\hat{M}^{-1/2}\hat{F}_x\hat{M}^{-1/2})+(\hat{M}^{-1/2}\hat{F}_y\hat{M}^{-1/2}\otimes I) \\ &= (I\otimes B)+(A\otimes I)\end{aligned} \tag{4.2}$$

where both A and B are diagonalizable. Thus $A = S\Lambda S^{-1}$, $B = TVT^{-1}$ and

$$\tilde{F} = (S\otimes T)(\Lambda\otimes I + I\otimes V)(S^{-1}\otimes T^{-1}) \tag{4.3}$$

so that

$$\tilde{F}^{-1} = (S\otimes T)(\Lambda\otimes I + I\otimes V)^{-1}(S^{-1}\otimes T^{-1}). \tag{4.4}$$

Since the transformed matrix $\tilde{F}$ can be diagonalized, the action of the inverse of F can be inexpensively applied as

$$\bar{F}^{e\,-1} = (\hat{M}^{-1/2}\otimes\hat{M}^{-1/2})(S\otimes T)(\Lambda\otimes I + I\otimes V)^{-1}(S^{-1}\otimes T^{-1})(\hat{M}^{-1/2}\otimes\hat{M}^{-1/2}). \tag{4.5}$$

This formulation only depends on the inverses of diagonal matrices, and of small matrices corresponding to one-dimensional phenomena. We exploit this in the application of our solvers and preconditioners as discussed in the following sections.

5. Preconditioning Advection Diffusion Equations via Iterative Substructuring. Applying a preconditioner P_F that cheaply retains the spectral properties of $F(\vec{w})$ one may significantly reduce the number of iterations needed to solve (3.8). To perform matrix-vector products involving P_F^{-1}, splitting methods based on multigrid have been applied successfully within a finite element framework [2]. However, we wish to maintain a matrix-free implementation based on local tensor products, so standard splitting techniques for F are not available. To retain the accelerated convergence of the Krylov subspace methods obtained via splitting, we instead our preconditioner using element-wise approximations of $F(w)$ by approximating the advection speed with a constant wind $\vec{w}^e$ on each element. That is, $P_F := \bar{F}$, where $\bar{F}$ is defined on each element in (4.1).

Using this local wind approximation to formulate P_F, we apply P_F^{-1} by performing iterative substructuring. The Fast Diagonalization Method is employed in element-wise solves to obtain interior degrees of freedom, and elemental boundary conditions are determined by an iterative Schur complement solve on the elemental interfaces. This procedure is similar to the Additive Schwarz method used in solving the Poisson equation described in [8], but with non-overlapping subdomains. By eliminating dense interior degrees of freedom via Fast Diagonalization iterations on the interfaces are performed on a significantly smaller system. Thus, $P_F = \bar{F}$ provides an inexpensive preconditioner that approximates local flow structure. We outline the use of substructuring methods for application of $P_F - 1$ in section 6.

In flows where the wind coefficient $\vec{w}$ is bi-constant, P_F^{-1} constitutes a direct solver to equation (3.8) (but with an iteration needed to apply $\bar{F}_0^{-1}$ on the union of element interfaces). We demonstrate the convergence and accuracy of this method as a direct solver in examples 7.1 and 7.2. In more general flows, P_F^{-1} is used as a preconditioner within an iterative solution

algorithm to solve equation (3.8). In such cases, the action of P_F^{-1} is performed as mentioned above at each step of the iterative scheme. Thus, the preconditioned iterative solver would require an inner iteration for the preconditioner at the interface nodes in (6.4) and an outer iteration for (3.8). In example 7.3 we demonstrate the use of P_F^{-1} as a preconditioner.

6. Domain Decomposition via Iterative Substructuring. In this section we outline the use of a matrix-free iterative substructuring method to perform the action of the preconditioner P_F^{-1}. Substructuring methods solve a PDE on a set of non-overlapping sub-domains by eliminating the interior degrees of freedom, solving for inter-element interfaces, and then using back substitution to solve for the interior degrees of freedom. In large systems, the degrees of freedom on elemental interfaces are determined using iterative methods. The system that governs the elemental interfaces may be poorly conditioned, and thus requires preconditioning. We outline the domain decomposition method that we use in conjunction with the Neumann-Neumann preconditioner for the interface solve.

Subdividing the domain into E spectral elements, with the elemental interfaces being represented by a set Γ, and interior degrees of freedom represented by sets I^e, one obtains a system of equations of the form

$$\begin{bmatrix} \bar{F}_{II}^1 & 0 & \dots & 0 & \bar{F}_{I\Gamma}^1 \\ 0 & \bar{F}_{II}^2 & 0 & \dots & \bar{F}_{I\Gamma}^2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \bar{F}_{II}^E & \bar{F}_{I\Gamma}^E \\ \bar{F}_{\Gamma I}^1 & \bar{F}_{\Gamma I}^2 & \dots & \bar{F}_{I\Gamma}^E & F_{\Gamma\Gamma} \end{bmatrix} \begin{pmatrix} u_{I^1} \\ u_{I^2} \\ \vdots \\ u_{I^E} \\ u_\Gamma \end{pmatrix} = \begin{pmatrix} b_{I^1} - \bar{F} u_B|_{I^1} \\ b_{I^2} - \bar{F} u_B|_{I^2} \\ \vdots \\ b_{I^E} - \bar{F} u_B|_{I^E} \\ b_\Gamma - \bar{F} u_B|_\Gamma \end{pmatrix} = \begin{pmatrix} \hat{b}_{I^1} \\ \hat{b}_{I^2} \\ \vdots \\ \hat{b}_{I^E} \\ \hat{b}_\Gamma \end{pmatrix}. \quad (6.1)$$

Boundary conditions are implemented outside of the system operator in (6.1), by subtracting $\bar{F} u_B$ from the right hand side vector. $\bar{F} u_B$ denotes the full advection-diffusion system matrix applied to the Dirichlet boundary vector u_B. The goal now is to solve for interface values u_Γ in order to perform back substitution and solve for u_{I^e} on each sub-domain interior. To solve for u_Γ the system matrix in (6.1) is split into a lower and upper part

$$\begin{bmatrix} I & 0 & \dots & 0 & 0 \\ 0 & I & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & I & 0 \\ \bar{F}_{\Gamma I}^1 \bar{F}_{II}^{1\,-1} & \bar{F}_{\Gamma I}^2 \bar{F}_{II}^{2\,-1} & \dots & \bar{F}_{\Gamma I}^E \bar{F}_{II}^{E\,-1} & I \end{bmatrix} \begin{bmatrix} \bar{F}_{II}^1 & 0 & \dots & 0 & \bar{F}_{I\Gamma}^1 \\ 0 & \bar{F}_{II}^2 & 0 & \dots & \bar{F}_{I\Gamma}^2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \bar{F}_{II}^E & \bar{F}_{I\Gamma}^E \\ 0 & 0 & \dots & 0 & F_0 \end{bmatrix} \quad (6.2)$$

where $\bar{F}_0 = \sum_{e=1}^E (\bar{F}_{\Gamma\Gamma}^e - \bar{F}_{\Gamma I}^e \bar{F}_{II}^{e\,-1} \bar{F}_{I\Gamma}^e)$ represents the Schur complement of the system. By multiplying both sides of (6.1) with the inverse of the lower triangular matrix, one obtains the system

$$\begin{bmatrix} \bar{F}_{II}^1 & 0 & \dots & 0 & \bar{F}_{I\Gamma}^1 \\ 0 & \bar{F}_{II}^2 & 0 & \dots & \bar{F}_{I\Gamma}^2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \bar{F}_{II}^E & \bar{F}_{I\Gamma}^E \\ 0 & 0 & \dots & 0 & F_0 \end{bmatrix} \begin{pmatrix} u_{I^1} \\ u_{I^2} \\ \vdots \\ u_{I^E} \\ u_\Gamma \end{pmatrix} = \begin{pmatrix} \hat{b}_{I^1} \\ \hat{b}_{I^2} \\ \vdots \\ \hat{b}_{I^E} \\ g_\Gamma \end{pmatrix} \quad (6.3)$$

with $g_\Gamma = \sum_{e=1}^{E}(\hat{b}_{\Gamma^e} - \bar{F}^e_{\Gamma I}\,\bar{F}^{e\,-1}_{II}\,\hat{b}_{I^e})$. The interface variables are then obtained by solving

$$\bar{F}_0 u_\Gamma = g_\Gamma. \tag{6.4}$$

Writing $\bar{F}_0 = \sum \bar{F}^e_0$ allows for efficient tensor product based (see appendix (8)) computation of the elemental matrix-vector products, which can be used to apply the matrix on each element inside a GMRES solver. Once u_Γ is obtained it is substituted back into (6.3) to provide elemental boundary conditions for the interior solves. The interior variables u_{I^e} are then obtained via element-wise fast diagonalization. We note that $\bar{F}^{e\,-1}_{II}$ in the Schur complement operator is performed in the same manner.

To reduce the number of GMRES iterations required to solve (6.4) a Neumann-Neumann preconditioner is employed. The Neumann-Neumann method uses $\sum D^{(e)^T}\left(\bar{F}^e_0\right)^{-1} D^{(e)}$ as a preconditioner for (6.4). A simple two-domain case illustrates that with this choice, the preconditioned system is approximately a scaled identity when the sub-domains are of similar size i.e. $\bar{F}^1_0 \approx \bar{F}^2_0$

$$(\bar{F}^{1\,-1}_0 + \bar{F}^{2\,-1}_0)(\bar{F}^1_0 + \bar{F}^2_0) = 2I + \bar{F}^{2\,-1}_0\bar{F}^1_0 + \bar{F}^{1\,-1}_0\bar{F}^2_0 = 2I + \bar{F}^{2\,-1}_0\bar{F}^1_0 + (\bar{F}^{2\,-1}_0\bar{F}^1_0)^{-1} \tag{6.5}$$

The matrices D^e are chosen to provide an appropriate inverse scaling factor. The convergence factor of the preconditioned system is bounded by $\frac{C}{H^2}(1+log(N))^2$ [7] where C grows with the norm of the advection coefficient $\|w\|$, N is the order of the spectral element basis functions and H is the diameter of a typical element Ω_e. In the case of many domains, it is essential to provide an additional coarse grid solve to eliminate the $1/H^2$ dependence. When applying the preconditioner it is not necessary to form $\bar{F}^{(e)\,-1}_0$ on each element since

$$\bar{F}^{(e)\,-1}_0 v = (0 \quad I)\,\bar{F}^{(e)\,-1}_{DD}\begin{pmatrix}0\\ I\end{pmatrix} v \tag{6.6}$$

where

$$\bar{F}^{(e)}_{DD} = \begin{bmatrix} \bar{F}^e_{II} & \bar{F}^e_{I\Gamma} \\ \bar{F}^e_{\Gamma I} & \bar{F}^e_{\Gamma\Gamma} \end{bmatrix} \tag{6.7}$$

7. Results. We demonstrate the effectiveness of a constant coefficient approximation on subdomains. First we use use fast diagonalization and domain decomposition as a solver for two cases (7.1, 7.2) where the advection coefficient is constant in each direction. We then demonstrate in section 7.3 how this method performs as a preconditioner for systems with non-constant advection coefficients.

7.1. Example 1: Analytic solution with outflow boundary layer. We first introduce a problem whose solution exhibits a dramatic change in the outflow boundary at $y = 1$. This problem has an analytic solution [3], and Table 7.1 shows that as the polynomial degree is doubled the discretization error is reduced by two orders of magnitude. This reduction in error confirms the expected exponential convergence to the solution. Additionally, Table 7.1 lists the number of GMRES iterations required to solve (6.4) for the velocity at the elemental interface nodes. A plot of the numerical solution and the velocity contours is given in Figure 7.1. By way of contrast, in Table 7.2 we show algebraic convergence results for a second

order finite element solution using P_F^{-1} as a solver. We mention that the number of iterations for the finite element method is greater due to the increased number of interface nodes. This substantiates a central efficiency of the spectral element method via the large volume to surface ratio of nodes on each element. In the discretization corresponding to Figure 7.1 there are roughly five times fewer interface nodes than there are interior nodes, this means the Fast Diagonalization Method can efficiently eliminate a large portion of the degrees of freedom directly, and we only need to obtain an iterative solution corresponding to a relatively small number of degrees of freedom, thereby significantly reducing the number of operations to obtain a solution.

TABLE 7.1
Spectral convergence for example 1.

N	$\|u-u_h\|_2$	Number of Iterations
4	1.185×10^{-1}	26
8	1.199×10^{-3}	39
16	1.308×10^{-5}	61
32	1.353×10^{-9}	132

TABLE 7.2
Algebraic convergence for example 1.

E	$\|u-u_h\|_2$	h	Number of Iterations
4	.4704	.25	50
8	.2605	.125	67
16	.0757	.0625	165
32	.0137	.03125	210

The iteration results in Tables 7.1 and 7.2 correspond to the number of iterates required to obtain an interface solution to (6.4) using non-preconditioned GMRES. In addition to these iterations, a solution on the element interiors is obtained via Fast Diagonalization. These results are preliminary since our Neumann-Neumann preconditioner for the Schur compliment solve in (6.4) is not yet in place. We expect the number of iterations to be reduced substantially via this preconditioner. In particular, we hope to obtain a fixed iteration count via a coarse grid multilevel Neumann-Neumann preconditioner.

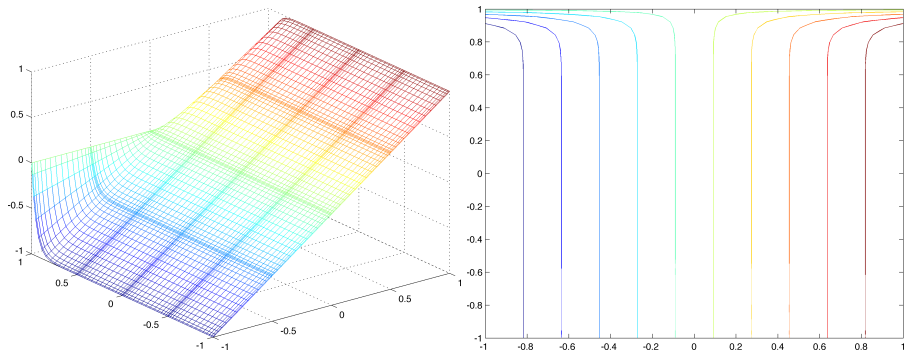

FIG. 7.1. *Steady advection diffusion flow with bi-constant winds. Left* $\vec{w}=(0,20)$ *and outflow boundary layer at* $y=1$*, convergence results in Table 7.1. Spectral element discretization with 4 elements in each dimension, and polynomial degree 16 on each element.*

7.2. Example 2: Moderate wind with internal boundary layer. In our second example we demonstrate the effectiveness of our method at capturing the internal boundary layer (see Figure 7.2) that results from a jump discontinuity in the inflow region of the boundary. In this flow the advection speed is significantly larger than in the previous example causing the flow to exhibit sharp features as shown in Figure 7.2. Using iterative substructuring, GMRES converges within 10^{-5} in 110 steps without Neumann-Neumann preconditioning. In this example, the Schur complement solve involves a system with roughly five times fewer unknowns than the global system.

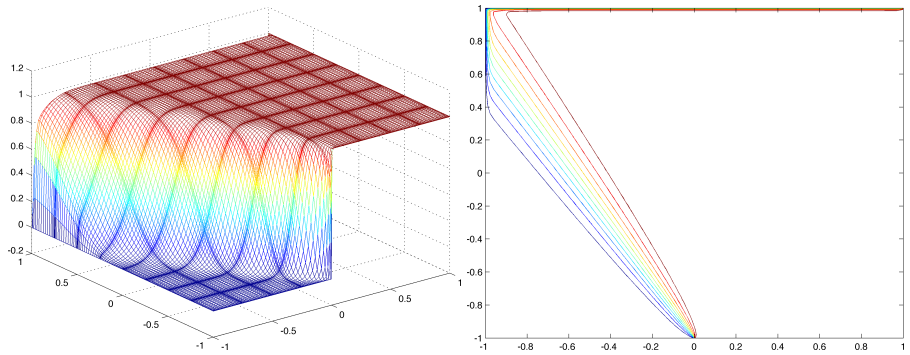

FIG. 7.2. *Steady advection diffusion flow with bi-constant winds* $\vec{w} = 200(-sin(\pi/6), cos(pi/6))$. *Flow exhibits an internal boundary layer* $O(\sqrt{1/200})$ *and jump discontinuity in boundary at (0,-1). Spectral element discretization with 8 elements in each dimension, and polynomial degree 16 on each element. GMRES converges to* 10^{-5} *in 110 steps without preconditioning.*

7.3. Example 3: Recirculating wind with characteristic boundary layers. To test our preconditioner, we use a recirculating wind $\vec{w} = 200(y(1-x^2), -x(1-y^2))$. Discontinuities at the corners of the nonzero boundary lead to boundary layers (see Figure 7.3). Figure 7.3 illustrates the convergence behavior of the preconditioned and non-preconditioned system for solving (3.8). Flexible GMRES [6] is used for the outer iteration, and non-preconditioned GMRES is used to solve the Schur complement system inexactly inside the application of the preconditioner. In Figure 7.3 we compare the number of iterations required for our preconditioned system; it is prohibitively expensive to acquire an accurate solution using GMRES without preconditioning, whereas the preconditioned system, using Flexible GMRES converges in 22 iterations. That is, we compute an inexact interface solution u_Γ with 1% accuracy, $\|\bar{F}_0 u_\Gamma - g_\Gamma\| < .1\|g_\Gamma\|$, this requires approximately 30 inner GMRES iterations. Because of this inexact inner step, we use Flexible GMRES [6] for the outer iteration.

By employing a Neumann-Neumann preconditioner for the Schur complement solve, we expect a reduced iteration count for the interface solve, and hope to be able to efficiently obtain a more accurate solution at the interface to improve the iteration count for the Flexible GMRES system. However these preliminary results are encouraging. The cost of the inexact interface solve is $O((N+1)(120E+N+1))$. Additionally each application of P_F^{-1} requires the Fast Diagonalization Method to be performed on element interiors the cost of this is roughly the same as a matrix-vector multiply $O(E(N+1)^2)$. The work per iteration will change with the inclusion of a Neumann-Neumann interface preconditioner, the number of operations required per application of P_F^{-1} will be $O(2E(N+1)^2 + (N+1)(4ME+N+1))$ where M denotes the number of interface iterations (30 in this example without the Neumann-Neumann operator), the additional factor of 2 in the first term corresponds the application of the Neumann-Neumann preconditioner as prescribed in equation (6.6).

8. Conclusion. We have developed a matrix-free solution method for tensor-product based discretizations of the steady advection-diffusion equation with bi-constant wind speed. This method is based on iterative substructuring, and uses Fast Diagonalization to eliminate interior degrees of freedom on each element. We have also shown how to use this method as a preconditioner for advection-diffusion systems with non-constant wind speeds. Preliminary results are positive, and we expect improved iteration counts by incorporating a Neumann-Neumann preconditioner into elemental interface solves.

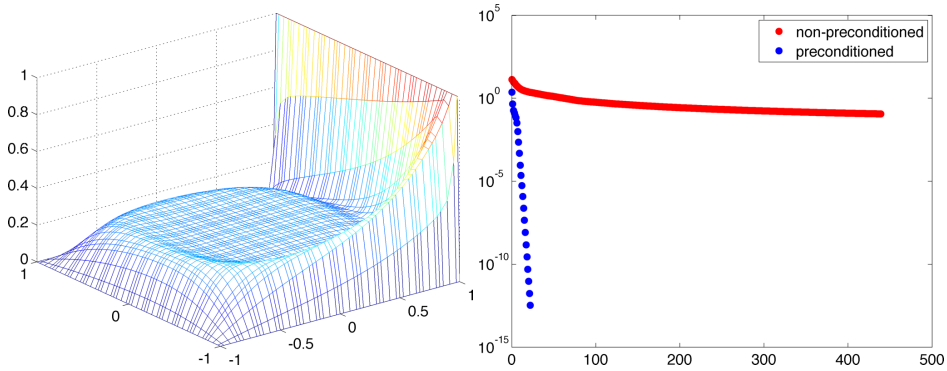

FIG. 7.3. *Computed solution of steady advection diffusion flow with recirculating wind* $\vec{w} = 200(y(1-x^2), -x(1-y^2))$. *A spectral element discretization with 12 elements in each dimension, and polynomial degree 4 on each element.(left) Comparison of preconditioned Flexible GMRES iterations and non-preconditioned GMRES iterations for solving equation (3.8). (right)*

Appendix: One Dimensional Matrix Decomposition for Matrix-Free Domain Decomposition. The tensor product basis of the spectral element method allows for efficient one dimensional dense matrix-matrix products to replace large sparse matrix-vector products. This tensor product formulation also allows for the use of the Fast Diagonalization Method, which is a key component of the element-based matrix-free preconditioning strategy we advocate in this paper. Because it is common to use node orderings that enumerate interior degrees of freedom and then boundary degrees of freedom we show the 1D building blocks needed to formulate the 2D operators in terms of their interior and boundary couplings within a lexigraphically ordered tensor product framework. We let N be the degree of the polynomial basis for a given discretization.

We write $\hat{F}_{(N+1)\times(N+1)}$ as the full 1D advection-diffusion matrix, and $\hat{B}_{(N+1)\times(N+1)}$ as the diagonal 1D mass matrix. $F_{(N+1)^2\times(N+1)^2} = \hat{F}\otimes\hat{B} + \hat{B}\otimes\hat{F}$ is the sparse 2D advection-diffusion matrix on a single element. We can decompose $\hat{F}$ and $\hat{B}$ into their interior and boundary couplings.

$\hat{F}_{ii} = \hat{F}(2:N, 2:N)$ Interior-Interior
$\hat{F}_{iLR} = \hat{F}(2:N, 1:N+1)$ Interior-Boundary
$\hat{F}_{iTB} = \hat{F}(1:N+1, 2:N)$ Boundary-Interior
$\hat{F}_{bb} = \hat{F}(1,1) + \hat{A}(1,N) + \hat{F}(N,1) + \hat{F}(N,N)$ Boundary-Boundary
$\hat{B}_{ii} = \hat{B}(2:N, 2:N)$ Interior-Interior
$\hat{B}_{bb} = \hat{B}(1:1, \vec{0}, N+1:N+1)$ Boundary-Boundary

This decomposition allows F to be written as $F = F_{II} + F_{\Gamma\Gamma} + F_{I\Gamma} + F_{\Gamma I}$ with

$F_{II} = \hat{F}_{ii}\otimes\hat{B}_{ii} + \hat{B}_{ii}\otimes\hat{F}_{ii}$
$F_{\Gamma\Gamma} = \hat{F}\otimes\hat{B}_{bb} + \hat{B}_{bb}\otimes\hat{F} + \hat{F}_{bb}\otimes\hat{B}_{ii} + \hat{B}_{ii}\otimes\hat{F}_{bb}$
$F_{I\Gamma} = \hat{F}_{iLR}\otimes\hat{B}_{ii} + \hat{B}_{ii}\otimes\hat{F}_{iLR}$
$F_{\Gamma I} = \hat{F}_{iTB}\otimes\hat{B}_{ii} + \hat{B}_{ii}\otimes\hat{F}_{iTB}$.

REFERENCES

[1] M. DEVILLE, P. FISCHER, AND E. MUND, *High-Order Methods for Incompressible Fluid Flows*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2002.
[2] H. ELMAN, V. HOWLE, J. SHADID, R. SHUTTLEWORTH, AND R. TUMINARO, *Block preconditioners based on approximate commutators*, SIAM J. Sci. Comput., 27 (2006), pp. 1651–1668.

[3] H. ELMAN, D. SILVESTER, AND A. WATHEN, *Finite Elements and Fast Iterative Solvers with applications in incompressible fluid dynamics*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 2005.

[4] R. LYNCH, J. RICE, AND D. THOMAS, *Direct solution of partial difference equations by tensor product methods*, Numer. Math., 6 (1964), pp. 185–199.

[5] Y. MADAY, A. PATERA, AND E. RØNQUIST, *An operator-integration-factor splitting method for time dependent problems: Application to incompressible fluid flow.*, Journal of Scientific Computation, (1990), pp. 263–292.

[6] Y. SAAD, *A flexible inner-outer preconditioned gmres algorithm*, SIAM Journal on Scientific Computing, 14 (1993), pp. 461–469.

[7] A. TOSELLI AND O. WIDLUND, *Domain Decomposition Methods - Algorithms and Theory*, Springer Series in Computational Mathematics, Springer, 2005.

[8] H. TUFO AND P. FISCHER, *Terascale spectral element algorithms and implementations*, in Supercomputing '99: Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM), New York, NY, USA, 1999, ACM Press, p. 68.