BLOCK PRECONDITIONERS FOR COUPLED PHYSICS PROBLEMS[*]

VICTORIA E. HOWLE[†], ROBERT C. KIRBY[‡], AND GEOFFREY DILLON[§]

Abstract. Finite element discretizations of multiphysics problems frequently give rise to block-structured linear algebra problems that require effective preconditioners. We build two classes of preconditioners in the spirit of well-known block factorizations [21, 16] and apply these to the diffusive portion of the bidomain equations and the Bénard convection problem. An abstract generalized eigenvalue problem allows us to give application-specific bounds for the real parts of eigenvalues for these two problems. This analysis is accompanied by numerical calculations with several interesting features. One of our preconditioners for the bidomain equations converges in five iterations for a range of problem sizes. For Bénard convection, we observe mesh-independent convergence with reasonable robustness with respect to physical parameters, and offer some preliminary parallel scaling results on a multicore processor via MPI.

1. Introduction.

Numerical discretization of systems of partial differential equations leads to challenging matrix equations with natural block structure. Iterative methods for such problems require effective preconditioners, which frequently make use of this block structure. Much of the early work on block preconditioners, such as that of Benzi, Golub, and Liesen [2], Elman and Silvester [5], and Elman, Howle, Shadid, Shuttleworth, and Tuminaro [6] focused on the Navier-Stokes equations. Our interest is in extending such methodology to other coupled systems such as the bidomain equations and Bénard convection. The bidomain equations are described in Keener and Sneyd [18]. Examples of solver methodology for the bidomain equations include the work of Mardal, Nielsen, Cai, and Tveito [20]. The Bénard convection problem and current solution methods are described in Carey and Oden [4], Ferziger and Perić [8], and Gresho and Sani [11].

All of the problems we consider give rise to a linear system of the general block form

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}. \tag{1.1}$$

With this block structure, we consider certain natural block preconditioners. A major advantage of algebraic approaches is that they are relatively simple to adapt to different kinds of problems. By considering two quite different problems, we give some idea of our methods' flexibility and also show how to adapt existing solvers for use in coupled problems.

For one, we consider a block diagonal (i.e., Jacobi) preconditioner

$$\mathbf{P}_J = \begin{bmatrix} A & 0 \\ 0 & D \end{bmatrix} \tag{1.2}$$

and block tridiagonal (i.e., Gauss-Seidel),

$$\mathbf{P}_{GS} = \begin{bmatrix} A & B \\ 0 & D \end{bmatrix}. \tag{1.3}$$

[*]This work was supported in part by NSF grant CCF-1117794.

[†]Department of Mathematics and Statistics; Texas Tech University; MS 1042; Lubbock, TX 79409-1042, victoria.howle@ttu.edu.

[‡]Department of Mathematics; Baylor University; One Bear Place #97328; Waco, TX 76798-7328, Robert_Kirby@baylor.edu.

[§]Department of Mathematics and Statistics; Texas Tech University; MS 1042; Lubbock, TX 79409-1042, geoffrey.dillon@ttu.edu.

For symmetric problems, a blockwise symmetric Gauss-Seidel such as

$$\mathbf{P}_{SGS} = \begin{bmatrix} A & B \\ 0 & D \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & D \end{bmatrix}^{-1} \begin{bmatrix} A & B \\ 0 & D \end{bmatrix}^{T}$$

is also possible, but we do not consider such a preconditioner in this work.

The block Jacobi and block Gauss-Seidel preconditioners may be viewed as cases of the preconditioners of Ipsen [16] that generalize the result of Murphy, Golub, and Wathen [21],

$$\mathbf{P} = \begin{bmatrix} A & 0 \\ 0 & -S \end{bmatrix}, \quad \text{and} \quad \mathbf{P} = \begin{bmatrix} A & B \\ 0 & S \end{bmatrix},$$

where S is the Schur complement $S = D - CA^{-1}B$. While Murphy, Golub, and Wathen assume that $D = 0$, Ipsen handles $D \neq 0$, which is required in our applications. The results in [16, 21] utilize S to provide a best-case result regarding eigenvalue bounds, but practical considerations require approximating S with something simpler. The preconditioners we consider amount to $D = S$. This was heuristically motivated by PDE theory in our previous work [14], and in this paper we obtain some application-specific eigenvalue bounds.

In particular, we consider the bidomain equations and Bénard convection problem, both leading to the general block structure of (1.1), but with different assumptions on the specific form of the subblocks A, B, C, and D. The bidomain equations, which model electrical activity in the heart, have that A and D are both symmetric and positive-definite, $B = C$, and B and C are both symmetric. We should point out that the bidomain equations do not quite fall into the standard category of saddle-point systems. Mardal, Nielsen, Cai, and Tveito [20] give theoretical and numerical results for both Jacobi and symmetric Gauss-Seidel preconditioners for this problem. For this problem, we focus on the nonsymmetric Gauss-Seidel preconditioner, which in our current numerical experiments converges in five iterations and has a provably better eigenvalue estimate than for block Jacobi.

We also consider the Bénard convection problem in which the Navier-Stokes equations are coupled to a scalar convection-diffusion problem. This system lacks the off-diagonal symmetry since we do not have $C = B^T$, nor do we have the symmetry of the diagonal blocks. This considerably complicates the analysis. We have given a heuristic justification of the block triangular preconditioner (1.3) for this problem in our previous work [14], where numerical results indicate it may be quite effective.

The differing assumptions on the form of the subblocks play a role in the analysis of the preconditioner, but they do not effect the form of the preconditioner itself. We view it as a feature of the algebraic approach that the preconditioner can be applied successfully to two such different applications.

The rest of this paper is organized as follows. We present more details of the bidomain problem and the Bénard convection problem and their finite element discretizations in Section 2. Then, we present eigenvalue analyses for the block Jacobi and block Gauss-Seidel preconditioners in Section 3, where we also relate these bounds to analytic properties of the problems and also compare to known results. Finally, we provide numerical results for our preconditioners in Section 4.

2. Target applications.

In this section, we describe the two target applications, the bidomain equations and Bénard convection, in more detail.

2.1. The bidomain equations. The bidomain system consists of a reaction-diffusion system of partial differential equations coupled to the well-known Fitzhugh–Nagumo equation. We assume that the reaction portion of the system (which has the interesting physics) is split off from the diffusion portion (which is the most expensive to calculate). Therefore we focus on the diffusion equations and effective preconditioners for the linear system arising from discretization using the finite element method in the spatial dimensions and the backward Euler method in time.

We are interested in computing the extracellular potential u_e and the transmembrane potential $v = u_i - u_e$ where u_i is the intracellular potential. There are two formulations of the problem, one for computing the pair (u_i, u_e) and one for (u_e, v). Like other authors ([20, 23]), we shall concentrate on the latter formulation since our methods perform substantially better on this formulation. Cardiac fibers are anisotropic by nature, a fact accounted for in the model by the conductivity tensors $M_i(\mathbf{x})$ and $M_e(\mathbf{x})$:

$$M_s(\mathbf{x}) = \sigma_t^s I + (\sigma_l^s - \sigma_t^s)\mathbf{a}(\mathbf{x})\mathbf{a}(\mathbf{x})^T, \quad s = i, e$$

where $\mathbf{a}(\mathbf{x})$ is the unit vector tangent to fiber at $\mathbf{x} \in \Omega \subset \mathbb{R}^2$, I is the identity tensor, and σ_l^s, σ_t^s are conductivity constants (intracellular and extracellular) along and across fibers.

The (u_e, v) formulation of the problem is to find $v(\mathbf{x}, t), u_e(\mathbf{x}, t), x \in \Omega, t \in [0, T]$ such that:

$$\begin{aligned}
c_m \partial_t v - \mathrm{div} M_i \nabla v + I_{ion} &= \mathrm{div} M_i \nabla u_e + I_{app}, && \text{in } \Omega \times [0,T],\\
-\mathrm{div} M \nabla u_e &= \mathrm{div} M_i \nabla v, && \text{in } \Omega \times [0,T],\\
\mathbf{n}^T M_i \nabla v = 0, &\qquad \mathbf{n}^T M \nabla u_e = 0, && \text{on } \Gamma \times [0,T],\\
v(\mathbf{x}, 0) = 0, &\qquad \text{in } \Omega.
\end{aligned}$$

Here, I_{app} is the current applied to begin the cardiac excitation process, I_{ion} is a cubic polynomial of v, and c_m is the surface capacity of the membrane. Assuming that the reaction and diffusion portions of the system are separated by operator splitting, the computationally dominant step is solving the coupled diffusion system whose weak form is given by

$$\begin{aligned}
(v^n, w_i) + (M_i \nabla v^n, \nabla w_i) + \Delta t\ (M_i\ \nabla u_e, \nabla w_i) &= (g_1, w_i)\\
(M \nabla u_e, \nabla w_e) + (M_i \nabla v^n, \nabla w_e) &= (g_2, w_e)
\end{aligned} \tag{2.1}$$

where w_i and w_e are test functions, $M = M_i + M_e$, and $(\cdot, \cdot)$ represents the traditional L^2 inner product. Discretization leads to a positive semidefinite system matrix of the form

$$\mathbf{A} = \begin{bmatrix} K_i + \frac{1}{\Delta t} M & K_i \\ K_i & K_e + K_i \end{bmatrix},$$

where M is given by

$$M = \int_\Omega \phi_i \phi_j \ dx$$

and K_s by

$$K_s = \sum_{T \in \mathcal{T}^h} \int_T (\nabla \phi_i)^T M_s \nabla \phi_j \, dx$$

for $s = i, e$, and where $\mathcal{T}_h$ is a triangulation of Ω (see [3]). Consequently, for this system, the subblocks in our general block matrix (1.1) are

$$\begin{aligned} A &= K_i + \frac{1}{\Delta t} M \\ B = C &= K_i \\ D &= K_i + K_e \end{aligned}$$

where A and D symmetric, with A positive definite and D semidefinite and B, C are symmetric. When we use D^{-1}, we refer to solution modulo the single-dimensional null space. In practice, we have added a slight regularization term to D to avoid the numerical intricacies of handling the null space.

2.2. Bénard convection. The Bénard convection problem consists of the incompressible Navier–Stokes equations coupled to a temperature convection-diffusion equation. The incompressible Navier–Stokes equations are given by:

$$\begin{aligned} -\nu \Delta u + u \cdot \nabla u + \nabla p &= f \\ \nabla \cdot u &= 0, \end{aligned}$$

where u and p are the fluid velocity and pressure and ν is the fluid viscosity. They are posed on some domain $\Omega \subset \mathbb{R}^d$ for $d = 2, 3$ and equipped with appropriate boundary conditions.

Coupling these equations to a temperature convection-diffusion equation, we get the Bénard convection equations:

$$\begin{aligned} -\Delta u + u \cdot \nabla u + \nabla p &= -\frac{Ra}{Pr} \hat{g} T \\ \nabla \cdot u &= 0 \\ -\frac{1}{Pr} \Delta T + u \cdot \nabla T &= 0, \end{aligned} \tag{2.2}$$

again posed on some domain Ω along with boundary conditions. The fluid velocity and pressure are again u and p, and the temperature is T. The Rayleigh number Ra measures the ratio of energy from buoyant forces to viscous dissipation and heat conduction, the Prandtl number Pr measures the ratio of viscosity to heat conduction, and $\hat{g}$ denotes a unit vector along the axis in which gravity acts. This model employs the Boussinesq approximation, in which temperature-dependent density variations are assumed to affect the momentum balance only through a buoyant force. For more information, including the non-dimensionalization used in (2.2), see [4, 8, 11]. We refer to the treatment of Carey and Oden [4] for more details. Note that, in this nondimensionalization, the effective Reynolds number is one, but large Rayleigh numbers can still lead to high fluid velocities.

We consider stabilized equal-order $P^1 - P^1$ discretizations of the fluid equations [15], and P^1 approximation of the temperature. We choose $P^1 - P^1$ because it has fewer degrees of freedom per cell for the velocity. There are 10×3 velocity degrees of freedom per cell for quadratics, but only 4×3 for linears. Hence, $P^1 - P^1$ is cheaper than Taylor-Hood. We let V_h be the vector-valued P^1 velocity space and W_h the scalar-valued P^1 pressure space. The stabilized thermal convection problem admits the weak form

$$\begin{aligned}(\nabla u_h, \nabla v_h) + (u_h \cdot \nabla u_h, v_h) - (p_h, \nabla \cdot v_h) &= \frac{Ra}{Pr}(T_h, v_{h,g}) \\ (\nabla \cdot u_h, w_h) + \beta h^2 (\nabla w_h, \nabla p_h) &= 0 \\ \frac{1}{Pr}(\nabla T_h, \nabla r_h) + (u_h \cdot \nabla T_h, r_h) &= 0,\end{aligned}$$

where $v_{h,g}$ means the component of v_h in the direction of gravity. Here, the term $\beta h^2 (\nabla w_h, \nabla p_h)$ is the stabilization term, where β is a stabilization parameter and h is the cell diameter.

Our numerical simulations will use a more complicated weak form based on Nitsche boundary conditions [1, 9, 22], in which the boundary conditions are imposed weakly via extra terms in the bilinear form rather than by extra constraint equations. Standard row-replacement Dirichlet boundary conditions disturb the block structure, while Nitsche boundary conditions leave the block structure intact and render self-adjoint bilinear forms as symmetric matrices.

For simplicity of exposition, however, we use standard Dirichlet boundary conditions in this paper, although the formulation naturally carries over to Nitsche-type conditions.

We consider a Newton-type linearization for the nonlinear system of equations. We linearize at the level of the weak form, leading to a linear variational problem for the Newton step. The weak form of the Jacobian is:

$$\begin{aligned}(\nabla u_h, \nabla v_h) + \left(u_h^0 \cdot \nabla u_h, v_h\right) + \left(u_h \cdot \nabla u_h^0, v_h\right) - (p_h, \nabla \cdot v_h) &= \frac{Ra}{Pr}(T_h, v_{h,g}) \\ (\nabla \cdot u_h, w_h) + \beta h^2 (\nabla w_h, \nabla p_h) &= 0 \\ \frac{1}{Pr}(\nabla T_h, \nabla r_h) + \left(u_h^0 \cdot \nabla T_h, r_h\right) + \left(u_h \cdot \nabla T_h^0, r_h\right) &= 0,\end{aligned}$$

for Bénard convection, where T_h^0 and u_h^0 are the temperature and velocity about which the system is linearized.

For the Navier–Stokes problem, this linearization and discretization gives rise to the following linear system for a single Newton step:

$$\begin{bmatrix} F & B^T \\ -B & R \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}, \tag{2.3}$$

where B and B^T are rectangular matrices corresponding to discrete divergence and gradient operators, F operates on the discrete velocity space, and R corresponds to the stabilization term.

The linear system for a Newton step for the stabilized convection problem takes the form

$$\begin{bmatrix} F & B^t & M_1 \\ -B & R & 0 \\ M_2 & 0 & K \end{bmatrix} \begin{bmatrix} u \\ p \\ T \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}.$$

The matrices F, B, and R are the same as in linearized Navier–Stokes. The matrix M_1 arises from the term $(\frac{Ra}{Pr}T, v_g)$, where g is the Cartesian direction in which gravity acts (y in 2d, z in 3d). In 2d, if the velocity variables in each direction are segregated, this has the form of $M_1^t = (0, M^t)$, where M is a rectangular mass matrix with rows corresponding to the P^2 basis functions and columns to P^1. The matrix M_2 arises from the Jacobian term $(u \cdot \nabla T_0, r)$, where T_0 is the temperature in the current Newton iterate. The matrix K comes from $\frac{1}{Pr}(\nabla T, \nabla r) + (u_0 \cdot \nabla T, r)$, so is a standard linear convection-diffusion operator. In the simulations in this paper, the fluid velocity has not become large enough to require additional stabilization of this term, although it would not complicate the block structure if such terms were included.

If we group together the velocity and pressure vectors into a single vector x, the convection system becomes

$$\begin{bmatrix} N & \widetilde{M_1} \\ \widetilde{M_2} & K \end{bmatrix} \begin{bmatrix} x \\ T \end{bmatrix} = \begin{bmatrix} g \\ f_3 \end{bmatrix},$$

where $\widetilde{M_1} = \begin{bmatrix} M_1 \\ 0 \end{bmatrix}$, $\widetilde{M_2} = [M_2, 0]$, and N represents the Navier–Stokes system in (2.3).

3. Analysis. In this section, we first set up generalized eigenvalue problems for the block diagonal and block triangular preconditioners without any specific assumptions about the subblocks in the general linear system. We then specialize the results to the specific cases represented in the bidomain equations and the Bénard convection problem. We include a graphical illustration of the preconditioner clustering the eigenvalues away from zero in each case.

3.1. Generalized Eigenvalue Problems. We now analyze the generalized eigenvalue problems for two types of block preconditioners: block diagonal (1.2) and block triangular (1.3).

The generalized eigenvalue problem for the block diagonal preconditioner is:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} A & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix},$$

which translates to

$$\begin{aligned} Ax + By &= \lambda Ax \\ Cx + Dy &= \lambda Dy. \end{aligned} \tag{3.1}$$

If $\lambda = 1$, we need $x \in$ the nullspace of B and $y \in$ the nullspace of C. Otherwise, if $\lambda \neq 1$, solving for x we get $x = \frac{1}{\lambda - 1} A^{-1} By$, which when plugged into (3.1) gives us

$$CA^{-1}By = (\lambda - 1)^2 Dy. \tag{3.2}$$

We also note that when the block diagonal preconditioner is applied to the generic system matrix, we get

$$\mathbf{A}\mathbf{P}_D^{-1} = \begin{bmatrix} I & BD^{-1} \\ CA^{-1} & I \end{bmatrix}. \tag{3.3}$$

The generalized eigenvalue problem for the block triangular preconditioner is:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} A & B \\ 0 & D \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix},$$

which translates to

$$\begin{aligned}Ax + By &= \lambda Ax + \lambda By\\ Cx + Dy &= \lambda Dy\end{aligned}$$

yielding an equation of the form

$$CA^{-1}By = (1-\lambda)Dy. \tag{3.4}$$

Applying the block triangular preconditioner to $\mathbf{A}$, we get:

$$\mathbf{A}\mathbf{P}_T^{-1} = \begin{bmatrix} I & 0 \\ CA^{-1} & I - CA^{-1}BD^{-1}\end{bmatrix}, \tag{3.5}$$

which means the eigenvalues of $\mathbf{A}\mathbf{P}_T^{-1}$ are 1 and $1-\lambda$, where λ represents the eigenvalues of $CA^{-1}BD^{-1}$.

It is worth noting that both generalized eigenvalue problems are of the form

$$CA^{-1}By = \mu Dy,$$

where $\mu = (\lambda-1)^2$ in the block diagonal case or $\mu = 1-\lambda$ in the block triangular case.

3.2. Application-specific Eigenvalue Bounds. In the next sections we specialize our generic eigenvalue bounds for the bidomain equations and the Bénard convection problem.

3.2.1. Bidomain equations. Mardal et al. [20, p. 90] obtain the following general result for the block diagonal preconditioner and apply it to the bidomain system:

THEOREM 3.1. *Under the assumptions that $C = B^T$ and that $\exists\,\alpha \in (0,1)$ such that*

$$2|(Bv,u)| \leq \alpha((Au,u) + (Dv,v)) \;\forall v,u \in \mathbb{R}^k, \tag{3.6}$$

we have that:

$$\kappa(\mathbf{A}\mathbf{P}_D) \leq \frac{1+\alpha}{1-\alpha}$$

where κ denotes the ratio of the maximum real parts of the eigenvalues of a given matrix to the minimum real parts of the eigenvalues.

The assumption (3.6) is proved for the bidomain equations in [20] on pages 92-93. With this assumption, we can show a stronger result for the block triangular preconditioner.

THEOREM 3.2. *Under the same assumptions as above, we have that*

$$\kappa(\mathbf{A}\mathbf{P}_T) \leq \frac{1+\alpha}{1-\alpha}\cdot\frac{1-\frac{\alpha}{2}}{1+\frac{\alpha}{2}}.$$

Proof. The technique here is similar to that in Mardal et al. [20]. We need to find constants c_0 and c_1 such that

$$\begin{aligned}c_0((Au,u) + (Bv,u) + (Dv,v)) &\leq (Au,u) + 2(Bv,u) + (Dv,v)\\ &\leq c_1((Au,u) + (Bv,u) + (Dv,v)).\end{aligned}$$

To begin, write

$$\begin{aligned}u^TAu + u^TBv + v^TDv &= u^TAu + 2u^TBv + v^TDv - u^TBv\\ &= u^TAu + 2u^TBv + v^TDv - xu^TBv - (1-x)u^TBv\end{aligned}$$

for some $x \in (0,1)$. Then, by using assumption (3.6) we get that

$$\begin{aligned}u^TAu + u^TBv + v^TDv &\geq u^TAu + 2u^TBv + v^TDv\\ &\qquad - \left[\frac{\alpha x}{2}(u^TAu + v^TDv) + (1-x)u^TBv\right]\\ &= (u^TAu + 2u^TBv + v^TDv)\left(1 - \frac{\alpha}{2(\alpha+1)}\right)\\ &= (u^TAu + 2u^TBv + v^TDv)\frac{\frac{\alpha}{2}+1}{\alpha+1}\end{aligned}$$

and so we have $c_1 = \frac{\alpha+1}{\frac{\alpha}{2}+1}$. Similarly we get that $c_0 = \frac{1-\alpha}{1-\frac{\alpha}{2}}$ and since generically

$$\kappa(\mathbf{AP}_T) \leq \frac{c_1}{c_0},$$

the result follows. □

Next we examine graphically the eigenvalues clustering from the preconditioner. In figure 3.1 we show eigenvalues from the unpreconditioned system (blue x's) and generalized eigenvalues from the preconditioned system (red circles). The domain is the unit square divided into an $N \times N$ grid, with $N = 16$ and $N = 32$; each square is subdivided into two right triangles. The conductivity constants used are $\sigma_l^e = 2.5 \times 10^{-3}$, $\sigma_t^e = 1.25 \times 10^{-3}$, $\sigma_l^i = 2.0 \times 10^{-3}$, $\sigma_t^i = 4.16 \times 10^{-4}$, and the timestep is $\tau = 4.0 \times 10^{-2}$. These values are taken from [23]. Figure 3.2 shows just the preconditioned eigenvalues for the same problems and preconditioners.

In Figures 3.1 and 3.2 the left two subfigures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two subfigures on the right show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$. The eigenvalues and generalized eigenvalues were obtained using MATLAB's *eig* function.

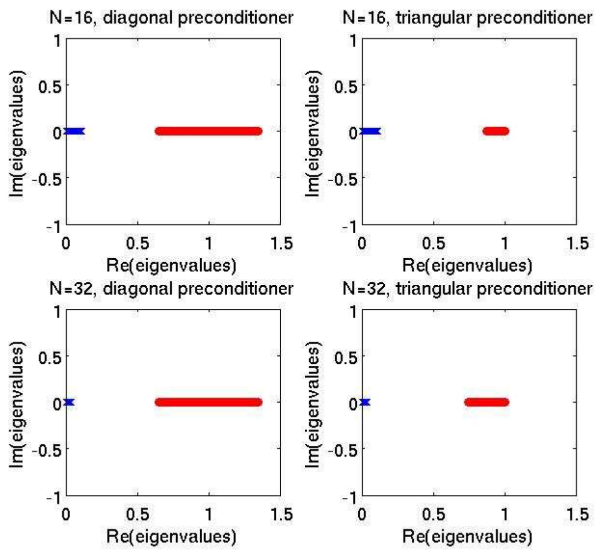

FIG. 3.1. *The left subfigures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two right subfigures show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$. Unpreconditioned eigenvalues are shown as blue x's and generalized eigenvalues from the preconditioned system are shown as red circles.*

Both preconditioners move the eigenvalues away from zero, but we can see that the triangular preconditioner clusters the eigenvalues somewhat better.

3.2.2. Bénard convection. We shall now estimate the size of the real parts of the eigenvalues of the problems (3.2) and (3.4) for the Bénard convection problem. We define $X_h = V_h \times W_h$ to be the product of the velocity and pressure/temperature spaces. The Jacobian operator of the equations can be written abstractly as

$$\begin{bmatrix} \mathcal{A} & \mathcal{B} \\ \mathcal{C} & \mathcal{D} \end{bmatrix}$$

where $\mathcal{A} : X_h \to X_h'$, $\mathcal{B} : W_h \to X_h'$, $\mathcal{C} : X_h \to W_h'$, and $\mathcal{D} : W_h \to W_h'$ are given by

$$\begin{aligned} \langle \mathcal{A}(u_h, p_h), (v_h, w_h) \rangle =& (\nabla u_h, \nabla p_h) + \left(u_h^0 \cdot \nabla u_h, v_h\right) - (p_h, \nabla \cdot v_h) \\ &+ (\nabla \cdot u_h, w_h) + \beta h^2 (\nabla w_h, \nabla p_h) \\ \langle \mathcal{B} T_h, (v_h, w_h) \rangle =& -\frac{Ra}{Pr} (T_h, v_{h,g}) \\ \langle \mathcal{C}(u_h, p_h), r_h \rangle =& \frac{1}{Pr} \left(u_h \cdot \nabla T_h^0, r_h\right) \\ \langle \mathcal{D} T_h, r_h \rangle =& \frac{1}{Pr} (\nabla T_h, \nabla r_h) + \left(u_h^0 \cdot \nabla T_h, r_h\right) \end{aligned}$$

The matrices A, B, C, D forming the blocks of the system matrix correspond to choosing test and trial functions as bases for the various finite element spaces. We will use the correspondence between the matrices and variational forms to establish eigenvalue bounds.

In order to prove our estimates we will need two lemmas.

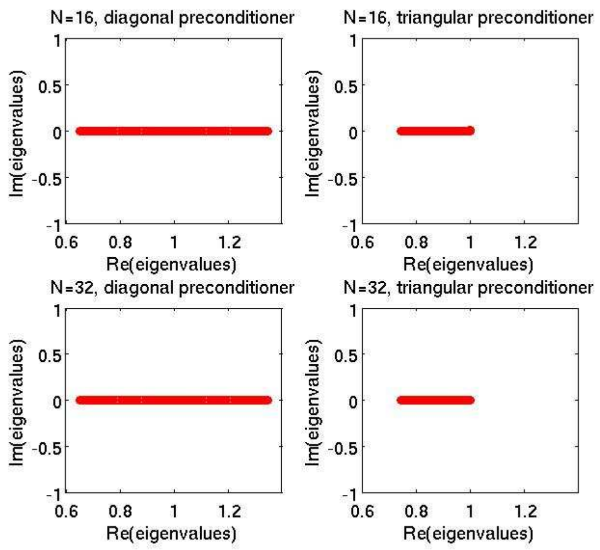

FIG. 3.2. *The left subfigures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two right subfigures on the right show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$.*

LEMMA 3.3. *Let D represent the $(2, 2)$ block of the Bénard convection problem and assume that $\nabla \cdot u_0 = 0$. Let $\{\phi_i\}_{i=1}^{\dim W_h}$ be a basis for W_h, $y \in \mathbb{R}^{\dim W_h}$ be arbitrary, and $r_h = \sum_{i=1}^{\dim W_h} y_i \phi_i$. Then*

$$y^T D y = \frac{1}{Pr} ||r_h||^2_{W_h}.$$

Proof. We have that

$$y^T D y = \langle \mathcal{D} r_h, r_h \rangle = \frac{1}{Pr} (\nabla r_h, \nabla r_h) + (u_0 \cdot \nabla r_h, r_h).$$

However if $\nabla \cdot u_0 = 0$ the bilinear form $(u_0 \cdot \nabla T_h, r_h) = -(u_0 \cdot \nabla r_h, T_h)$ is skew symmetric, so $(u_0 \cdot \nabla r_h, r_h) = 0$ and the result follows. □

This next lemma gives us a bound on the real parts of the eigenvalues of μ from (3.1).

LEMMA 3.4. *For y and r_h as in the previous Lemma, we have*

$$|y^T C A^{-1} B y| \leq (C_P)^4 \frac{Ra}{Pr} ||\nabla T_0^h||_\infty \cdot |y^T D y|$$

where C_P represents the constant from Poincaré's inequality, $||\nabla T_0^h||_\infty$ is the usual infinity norm.

Proof. We first need estimates for the $\mathcal{B}$ and $\mathcal{C}$ blocks. From the $\mathcal{B}$ block of (2.2) we have that

$$\begin{aligned}
|\langle \mathcal{B}r_h, (v_h, w_h)\rangle| &\le \frac{Ra}{Pr} \|v_h\|_{L^2} \|r_h\|_{L^2} \\
&\le \frac{Ra}{Pr} (C_P)^2 \|v_h\|_{W_h} \ \|r_h\|_{W_h} \\
&\le \frac{Ra}{Pr} (C_P)^2 \left(\|v_h\|_{W_h} + \|w_h\|_{L^2}\right) \|r_h\|_{W_h}
\end{aligned}$$

and so $\|\mathcal{B}\|_{\mathcal{L}(W_h, X_h')} \le \frac{Ra}{Pr}(C_P)^2$. From the $\mathcal{C}$ block of (2.2) we get

$$\begin{aligned}
|\langle \mathcal{C}(v_h, w_h), r_h\rangle| &\le ||v_h \cdot \nabla T_0^h||_{L^2} \ ||r_h||_{L^2} \\
&\le \ (C_P)^2 ||\nabla T_0^h||_\infty \ ||v_h||_{W_h} \ ||r_h||_{W_h} \\
&\le \ (C_P)^2 ||\nabla T_0^h||_\infty \ (||v_h||_{W_h} + ||w_h||_{L^2}) \ ||r_h||_{W_h}
\end{aligned}$$

which means $||\mathcal{C}||_{\mathcal{L}(X_h, W_h')} \le (C_P)^2 ||\nabla T_0^h||_\infty$. We now exploit with the following relationship:

$$y^T C A^{-1} B y = \left\langle r_h, \mathcal{C}\mathcal{A}^{-1}\mathcal{B} r_h \right\rangle$$

and break up the latter term as follows:

$$\begin{aligned}
\left\langle r_h, \mathcal{C}\mathcal{A}^{-1}\mathcal{B} r_h \right\rangle &\le \left\|\mathcal{C}\mathcal{A}^{-1}\mathcal{B}\right\|_{\mathcal{L}(W_h, W_h')} \ \|r_h\|_{W_h}^2 \\
&\le \|\mathcal{C}\|_{\mathcal{L}(X_h, W_h')} \left\|\mathcal{A}^{-1}\right\|_{\mathcal{L}(X_h', X_h)} \|\mathcal{B}\|_{\mathcal{L}(W_h, X_h')} \|r_h\|_{W_h}^2 \\
&\le \|\mathcal{C}\|_{\mathcal{L}(X_h, W_h')} \left\|\mathcal{A}^{-1}\right\|_{\mathcal{L}(X_h', X_h)} \|\mathcal{B}\|_{\mathcal{L}(W_h, X_h\prime)} \|r_h\|_{W_h}^2 \\
&= \|\mathcal{C}\|_{\mathcal{L}(X_h, W_h')} \left\|\mathcal{A}^{-1}\right\|_{\mathcal{L}(X_h', X_h)} \|\mathcal{B}\|_{\mathcal{L}(W_h, X_h')} \ y^T D y
\end{aligned}$$

by the previous lemma and the result follows. □

These two lemmas give bounds on $|\Re(\mu)|$, where μ is a generalized eigenvalue of (3.3), and we will use them to obtain bounds for the preconditioned systems later.

THEOREM 3.5. *The real parts of the eigenvalues of the Bénard convection system with a block diagonal preconditioner are given by*

$$|\Re(\lambda)| \le 1 + (C_P)^2 \sqrt{\frac{Ra}{Pr} \left\|\nabla T_0^h\right\|_\infty \|\mathcal{A}^{-1}\|_{\mathcal{L}(X_h', X_h)}}.$$

The $||\mathcal{A}^{-1}||_{\mathcal{L}(X', X)}$ term will have a mesh independent bound that depends on $||u_0||_{H_0^1}$ amongst others. For details see [15].

Proof.

By equation (3.2) and the previous lemma, we have

$$(\lambda - 1)^2 \le \ (C_P)^4 \frac{Ra}{Pr} \left\|\nabla T_0^h\right\|_\infty \left\|\mathcal{A}^{-1}\right\|_{\mathcal{L}(X_h', X_h)}$$

and so

$$|\Re(\lambda)| \le 1 + (C_P)^2 \sqrt{\frac{Ra}{Pr} \left\|\nabla T_0^h\right\|_\infty \ \|\mathcal{A}^{-1}\|_{\mathcal{L}(X_h', X_h)}}.$$

□

The eigenvalues of the preconditioned system (3.5) have the following bound:

THEOREM 3.6. *The eigenvalues of $CA^{-1}BD^{-1}$ for the Bénard convection problem are bounded by*

$$|\Re(\lambda)| \leq 1 + (C_P)^4 \frac{Ra}{Pr} \left\|\nabla T_0^h\right\|_\infty \left\|\mathcal{A}^{-1}\right\|_{\mathcal{L}(X_h', X_h)} \left\|\mathcal{D}^{-1}\right\|_{\mathcal{L}(W_h', W_h)} .$$

Proof. First, note that

$$\begin{aligned} |\lambda| &\leq 1 + \left\|\mathcal{C}\mathcal{A}^{-1}\mathcal{B}\mathcal{D}^{-1}\right\|_{\mathcal{L}(W_h')} \\ &\leq 1 + ||\mathcal{C}||_{\mathcal{L}(X_h, W_h')} \, ||\mathcal{A}^{-1}||_{\mathcal{L}(X_h', X_h)} \, ||\mathcal{B}||_{\mathcal{L}(W_h, X_h')} \\ &\quad ||\mathcal{D}^{-1}||_{\mathcal{L}(W_h', W_h)} . \end{aligned}$$

The result follows by now applying lemma 3.4.
□

Next we examine graphically the eigenvalue clustering from the preconditioner. The problem used here is a slight modification of the classic two-dimensional Bénard convection problem with fluid in a box with no-slip boundary conditions for the fluid. A unit temperature difference is imposed in the horizontal direction, and insulating boundary conditions are applied on the remaining sides.

In Figures 3.3 and 3.4, the left two subfigures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two subfigures on the right show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$. In both figures, the system has Rayleigh number 2×10^4.

In Figure 3.3, we show eigenvalues from the unpreconditioned system (blue x's) and generalized eigenvalues from the preconditioned system (red circles) on two dimensional problems with 16 and 32 nodes. Figure 3.4 shows just the preconditioned eigenvalues for the same problems and preconditioners. The eigenvalues and generalized eigenvalues were obtained using MATLAB's *eig* function.

Both preconditioners cluster the eigenvalues, but we can see that the triangular preconditioner leads to a tighter cluster, particularly in the imaginary axis. The diagonal preconditioner gives a more skew system and the triangular preconditioner a less skew system.

4. Numerical Results.

In this section we examine results for the block Jacobi and block Gauss-Seidel versions of the preconditioners on the bidomain equations and on the Bénard convection problem.

For our numerical calculations, we are using the Sundance library [19], which is part of the Trilinos framework [13], also making use of various Trilinos packages for the linear solves that arise. For the preconditioner to be efficient, we employ iterative methods with fairly low accuracy on the linear subsolves. This requires the use of flexible GMRES (FGMRES) [24], for the outer iteration, which we access through the package Belos [25].

4.1. Bidomain Results.

We begin with applying the two preconditioners to the bidomain equations. In each case, application of the preconditioner involves two linear subsolves. For the inner subsolves, we use GMRES with an algebraic multigrid preconditioner. We use the Trilinos package Belos [25] for GMRES with ML [10] for the algebraic multigrid.

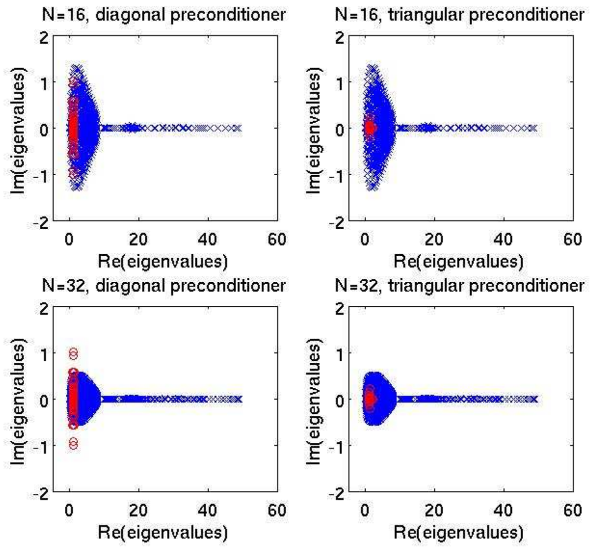

Fig. 3.3. *The left two figures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two figures on the right show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$. Eigenvalues from the unpreconditioned system are shown as blue x's and generalized eigenvalues from the preconditioned system are shown as red circles.*

We assume a convergence tolerance of 1×10^{-6} for both the inner and outer linear solves. The domain is the unit square divided into an $N \times N$ grid with each square subdivided into two right triangles. The conductivity constants used are $\sigma_l^e = 2.5 \times 10^{-3}$, $\sigma_t^e = 1.25 \times 10^{-3}$, $\sigma_l^i = 2.0 \times 10^{-3}$, and $\sigma_t^i = 4.16 \times 10^{-4}$, and the conductivity tensors are

$$M_i = \frac{1}{2}\begin{bmatrix} \sigma_l^i + \sigma_t^i & \sigma_l^i - \sigma_t^i \\ \sigma_l^i - \sigma_t^i & \sigma_l^i + \sigma_t^i \end{bmatrix}, M_e = \frac{1}{2}\begin{bmatrix} \sigma_l^e + \sigma_t^e & \sigma_l^e - \sigma_t^e \\ \sigma_l^e - \sigma_t^e & \sigma_l^e + \sigma_t^e \end{bmatrix}.$$

We use a time step of $\Delta t = 4 \times 10^{-2}$ as recommended in [23].

Table 4.1 reports outer iteration counts and solve times for the block Jacobi and block Gauss-Seidel preconditioners on a serial run. Iteration counts are given first, with timings, in seconds, given in parentheses. These calculations are performed on a Mac Pro desktop with dual quad-core 2.8 GHz Xeon processors with 32GB of RAM running OSX version 10.5 and using gcc 4.4.4 installed from the MacPorts system.

We can see in Table 4.1 that the block triangular preconditioner performs quite well, needing only five outer iterations regardless of the mesh. The dominant cost of this one outer iteration is the pair of variable-coefficient diffusion solves to comparable tolerance and one matrix-vector product that are required to apply the preconditioner. The block diagonal preconditioner, on the other hand, takes more outer iterations, but does seem to scale better on these problems in terms of timing.

4.2. Bénard Results. We consider a three-dimensional version of the coupled fluid-convection problem. In this problem, we have fluid in a three-dimensional box with no-slip boundary conditions for the fluid. We impose a temperature $T = 1$ on one face, $T = 0$ on the opposite face, and $\frac{\partial T}{\partial n} = 0$ on the remaining faces. At larger

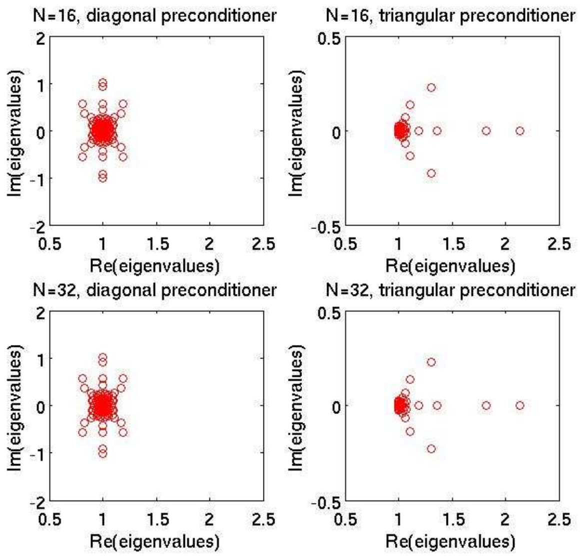

Fig. 3.4. *The left two figures show eigenvalues for the system with the diagonal preconditioner for $N = 16$ and $N = 32$. The two figures on the right show eigenvalues for the system with the triangular preconditioner for $N = 16$ and $N = 32$.*

N	Block Diagonal	Block Triangular
128	9 (2.957)	5 (1.614)
256	10 (15.95)	5 (7.622)
512	10 (76.48)	5 (37.09)
1024	10 (407.2)	5 (199.8)

Table 4.1

Bidomain problem comparing block diagonal and block triangular preconditioner: Number of FGMRES iterations when the linear system is solved using the inexact block diagonal and block triangular preconditioners. An FGMRES tolerance of 1×10^{-6} was used for the outer iteration, and GMRES with a tolerance 1×10^{-6} and preconditioned with algebraic multigrid was used for the preconditioner subsolves.

Rayleigh numbers, this creates an instability leading to overturning cells. We consider a range of Rayleigh numbers and problem sizes, labeled "cube-i" with numbers of vertices and tetrahedra as shown in Table 4.2.

The Bénard convection problem is a much more difficult problem than the bidomain equations. These calculations are performed on a Dell Precision desktop with dual 2.7 GHz eight-core Xeon processors and 128 GB of RAM running Linux Mint 13 and using gcc 4.6.3.

We applied both the block diagonal and block triangular preconditioners to this problem. In all cases, we used a simple Newton iteration to an outer Euclidean tolerance of 10^{-6} and preconditioned flexible GMRES with Euclidean tolerance of 10^{-8} for the linear solve. Both our preconditioners require solution of the underlying linearized Navier-Stokes and temperature equations. These were done with flexible GMRES, to begin with to a relatively tight tolerance of 10^{-6}. For the Navier-Stokes solve, we used the Pressure Convection–Diffusion (PCD) preconditioner [17, 6, 7].

Mesh	Vertices	Tetrahedra
cube-1	144	456
cube-2	855	3648
cube-3	5805	29184

TABLE 4.2

Numbers of vertices and tetrahedra for each of the four meshes used for Bénard convection simulations.

The PCD preconditioner requires solves on the convection-diffusion block F, on a pressure Laplacian matrix A_p, and on a pressure mass matrix M_p[1]. Within the PCD preconditioner, each subsolve was performed with GMRES (for F) and CG (for A_p and M_p) to a tolerance of 10^{-8}, preconditioned with algebraic multigrid. We use the Trilinos package Belos for the PCD linear systems and Aztec [12] for the K solve, with ML [10] for the algebraic multigrid.

NS tol = 10^{-6}, PCD tols = 10^{-8}, K tol = 10^{-8}						
Ra	2×10^2		2×10^3		2×10^4	
Mesh	NL	LIN (Time)	NL	LIN (Time)	NL	LIN (Time)
cube-1	4	6.5 (17.7)	5	9.8 (34.54)	7	16 (97.87)
cube-2	4	6.75 (73.47)	5	10 (148.5)	7	19 (513.3)
cube-3	3	6.33 (479.3)	5	10 (1483)	7	20.29 (5660)

TABLE 4.3

3D problem with block diagonal preconditioner: *Number of Newton steps and average FGMRES iteration count per Newton step when the linear system is solved using the block diagonal preconditioner. "NL" refers to the number of nonlinear iterations required to obtain a residual of* 10^{-6}*, "LIN" the average number of outer GMRES iterations per Newton step, and Time the total number of seconds spent in the solve. The number of Newton iterations and the average number of iterations per Newton step are essentially independent with some moderate dependence on Ra.*

NS tol = 10^{-6}, PCD tols = 10^{-8}, K tol = 10^{-6}						
Ra	2×10^2		2×10^3		2×10^4	
N	NL	LIN (Time)	NL	LIN (Time)	NL	LIN (Time)
cube-1	4	3.5 (10.59)	5	4.4 (17.25)	7	7.57 (50.85)
cube-2	4	3.33 (39.86)	5	4.8 (74.54)	7	8.57 (236.7)
cube-3	3	3.33 (269.5)	5	5 (735.5)	7	9.29 (2711)

TABLE 4.4

3D problem with block triangular preconditioner: *Number of Newton steps and average FGMRES iteration count per Newton step when the linear system is solved using the inexact block triangular preconditioner. "NL" refers to the number of nonlinear iterations required to obtain a residual of* 10^{-6}*, "LIN" the average number of outer GMRES iterations per Newton step, and Time the total number of seconds spent in the solve. The number of Newton iterations and the average number of iterations per Newton step are essentially independent with some dependence on Ra. The run-times and iteration counts are lower than for the block diagonal case.*

We can see that for both preconditioners, the number of Newton iterations and the

[1] Some gains in efficiency can be obtained by lumping masses or neglecting off-diagonal terms to obtain a trivially invertible approximation to M_p^{-1}. However, applying F^{-1} is far more expensive than M_p^{-1}, so this is a small practical gain that we neglect for the purposes of understanding our preconditioner

average number of FGMRES iterations per Newton step are essentially independent with respect to problems size. The number of Newton steps is mildly dependent on Ra, and the number of FGMRES iterations per Newton step grows more strongly with Ra as expected. The block triangular preconditioner seems to perform better than the block diagonal, both in terms of the run times and linear iteration counts. Although the times are quite large, we may interpret the overall cost as follows. For the block triangular preconditioner, the Newton iteration averaged less than ten linear solves to achieve convergence. As the dominant cost of each iteration is the linearized Navier-Stokes solve, this means that the coupled problem is, per Newton iteration, less than ten times the cost of Navier-Stokes. We interpret this as a very positive result, especially as any progress in preconditioning the Navier-Stokes equations can readily be incorporated into this strategy.

Using flexible GMRES, loosening the tolerances used in the solvers embedded in the preconditioners often offers some advantage as a significant reduction in the cost per iteration can outweights a moderate increase in iteration count. To investigate this for Bénard convection, we raised the FGMRES tolerance for the Navier-Stokes and temperature solutions to 10^{-2} and the tolerances inside the PCD preconditioner to 10^{-4}. We see in Tables 4.5 and 4.6 that this situation occurs for the block diagonal but not the block triangular iteration. The increased tolerances do lead to moderate growth in the number of linear iterations but a lower overall run time. However, the block triangular preconditioner degrades far more severely with the increased tolerances, leading to a net growth in run time. From these results, it seems that the block triangular preconditioner with a relatively tight tolerance seems to provide the best results.

NS tol = 10^{-2}, PCD tols = 10^{-4}, K tol = 10^{-2}						
Ra	2×10^2		2×10^3		2×10^4	
N	NL	LIN (Time)	NL	LIN (Time)	NL	LIN (Time)
cube-1	4	9.75 (16.57)	5	14.6 (32.32)	7	23 (92.66)
cube-2	4	9.75 (63.31)	5	15.6 (133.3)	7	26.71 (414.5)
cube-3	3	9.33 (358.9)	5	16 (1220)	7	29.43 (4202)

TABLE 4.5

3D problem with block diagonal preconditioner and relaxed inner solve tolerances: *Number of Newton steps and average FGMRES iteration count per Newton step when the linear system is solved using the inexact block diagonal preconditioner. "NL" refers to the number of nonlinear iterations required to obtain a residual of* 10^{-6}*, "LIN" the average number of outer GMRES iterations per Newton step, and Time the total number of seconds spent in the solve. The number of Newton iterations and the average number of iterations per Newton step are still essentially independent of the mesh with some dependence on Ra. Relative to the block-diagonal preconditioner with tighter tolerances, the linear iteration count increases and the total run-time decreases.*

As the block triangular preconditioner with moderately tight tolerances seemed to perform the best, we examined its parallel speedup by using one, two, four, and eight MPI jobs on the same multicore machine for the $Ra = 2.e4$ case on cube-3. In Table 4.7, we see that

Finally, in Table 4.7 we examine parallel speed up for the two block preconditioners. We show timings as we increase the number of MPI jobs (on a single multicore node) from 1 to 8 for the 3D Bénard problem with Rayleigh number 2×10^4 on the fine mesh *cube-3*. We see superlinear speedup observed for 2 and 4 processors is likely the result of better cache usage, but a speedup of 7.28 is still obtained using 8 cores, amounting to 90% parallel efficiency. This offers hope of scalability to larger machines.

NS tol = 10^{-2}, PCD tols = 10^{-4}, K tol = 10^{-2}						
Ra	2×10^2		2×10^3		2×10^4	
N	NL	LIN (Time)	NL	LIN (Time)	NL	LIN (Time)
cube-1	4	6.5 (10.07)	5	11 (16.22)	7	28.14 (58.74)
cube-2	4	6.75 (42.29)	6	12.4 (77.79)	7	32.42 (299)
cube-3	3	7 (259)	5	11.8 (668.1)	7	35 (3202)

Table 4.6

3D problem with block triangular preconditioner and relaxed inner solve tolerances: *Number of Newton steps and average FGMRES iteration count per Newton step when the linear system is solved using the inexact block diagonal preconditioner. "NL" refers to the number of nonlinear iterations required to obtain a residual of* 10^{-6}*, "LIN" the average number of outer GMRES iterations per Newton step, and Time the total number of seconds spent in the solve. The number of Newton iterations and the average number of iterations per Newton step are still essentially independent of the mesh with some dependence on Ra, but increasing the Navier-Stokes tolerances so increases the linear iteration count that the run-times increase.*

NP	total time (sec)	time per linear solve	speedup	efficiency
1	2711	387	1	1
2	1269	181	2.14	1.07
4	529	88.1	5.12	1.28
8	372	62	7.28	0.91

Table 4.7

Speedup obtained by parallelizing the Bénard convection problem with $Ra = 2e4$ *on multiple MPI processes of a single workstation using the block triangular preconditioner. The superlinear speedup observed for 2 and 4 processors is likely the result of better cache usage, but a speedup of 7.28 is still obtained using 8 cores, amounting to 90% parallel efficiency.*

5. Conclusions.

Based on a particular kind of Schur complement approximation, we have applied the framework of Ipsen [16] to two problems of contemporary interest, the bidomain equations and Bénard convection. Although much work remains, such as field-of-values analysis that would rigorously describe GMRES convergence theory, our eigenvalue analysis for Bénard convection seems to be the first such theoretical consideration in the literature. By reusing scalable solvers for the component problems, we are stepping toward a scalable solution strategy for multiphysics problems. Our block triangular preconditioner for the bidomain equations, empirically converging five iterations, requires a matrix-vector product and two variable-coefficient elliptic solves as its dominant cost. For Bénard convection, we are able to reuse scalable Navier-Stokes methodology such as the PCD preconditioner to obtain empirical mesh-independence, and MPI-based scalability on a multicore platform.

REFERENCES

[1] R. Becker, *Mesh adaptation for Dirichlet flow control via Nitsche's method*, Communications in numerical methods in engineering, 18 (2002), pp. 669–680.

[2] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numerica, 14 (2005), pp. 1–137.

[3] S. C. Brenner and L. R. Scott, *The mathematical theory of finite element methods*, vol. 15 of Texts in Applied Mathematics, Springer, New York, third ed., 2008.

[4] G. F. Carey and J. T. Oden, *Finite elements. Vol. VI*, The Texas Finite Element Series, VI, Prentice Hall Inc., Englewood Cliffs, NJ, 1986. Fluid mechanics.

[5] H. Elman and D. Silvester, *Fast nonsymmetric iterations and preconditioning for Navier-Stokes equations*, SIAM Journal on Scientific Computing, 17 (1996), pp. 33–46.

[6] H. C. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *Block preconditioners based on approximate commutators*, SIAM Journal on Scientific Computing, 27 (2006), pp. 1651–1668.

[7] H. C. Elman, V. E. Howle, J. Shadid, D. Silvester, and R. Tuminaro, *Least squares preconditioners for stabilized discretizations of the Navier–Stokes equations*, SIAM Journal on Scientific Computing, 30 (2007), pp. 290–311.

[8] J. H. Ferziger and M. Perić, *Computational Methods for Fluid Dynamics*, Springer, 1996.

[9] J. Freund and R. Stenberg, *On weakly imposed boundary conditions for second order problems*, in Proceedings of the Ninth Int. Conf. Finite Elements in Fluids, Venice, 1995, pp. 327–336.

[10] M. W. Gee, C. M. Siefert, J. J. Hu, R. S. Tuminaro, and M. G. Sala, *ML 5.0 smoothed aggregation user's guide*, Tech. Rep. SAND2006-2649, Sandia National Laboratories, 2006.

[11] P. M. Gresho and R. L. Sani, *Incompressible Flow and the Finite Element Method*, Springer, 1998.

[12] M. A. Heroux, *AztecOO user guide*, Tech. Rep. SAND2004-3796, Sandia National Laboratories, 2004.

[13] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, and K. S. Stanley, *An overview of the Trilinos project*, ACM Transactions on Mathematical Software, 31 (2005), pp. 397–423.

[14] V. E. Howle and R. C. Kirby, *Block preconditioners for finite element discretization of incompressible flow with thermal convection*, Numerical Linear Algebra with Applications, 19 (2012), pp. 427–440.

[15] T. J. Hughes, L. P. Franca, and M. Balestra, *A new finite element formulation for computational fluid dynamics: V. circumventing the Babuška–Brezzi condition: a stable Petrov–Galerkin formulation of the Stokes problem accommodating equal-order interpolations*, Computer Methods in Applied Mechanics and Engineering, 59 (1986), pp. 85–99.

[16] I. C. F. Ipsen, *A note on preconditioning nonsymmetric matrices*, SIAM Journal on Scientific Computing, 23 (2001), pp. 1050–1051.

[17] D. Kay, D. Loghin, and A. Wathen, *A preconditioner for the steady-state Navier–Stokes equations*, SIAM J. Sci. Comput., 24 (2002), pp. 237–256.

[18] J. Keener and J. Sneyd, *Mathematical Physiology*, Springer, New York, 1998.

[19] K. Long, *Large-Scale PDE-Constrained Optimization*, Lecture Notes in Computational Science and Engineering, Springer, Heidelberg, 2003, ch. Sundance: a rapid prototyping toolkit for parallel PDE simulation and optimization.

[20] K.-A. Mardal, B. F. Nielsen, X. Cai, and A. Tveito, *An order optimal solver for the discretized bidomain equations*, Numerical Linear Algebra with Applications, 14 (2007), pp. 83–98.

[21] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM Journal on Scientific Computing, 21 (2000), pp. 1969–1972.

[22] J. Nitsche, *On Dirichlet problems using subspaces with nearly zero boundary conditions*, The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations, (1972), pp. 603–627.

[23] M. Pennacchio and V. Simoncini, *Algebraic multigrid preconditioners for the bidomain reactiondiffusion system*, Applied Numerical Mathematics, 59 (2009), pp. 3033–3050.

[24] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.

[25] H. K. Thornquist, *Belos*, http://trilinos.sandia.gov/packages/belos/index.html.