A Comparison of Multilevel Solvers for the Cardiac Bidomain Equations

Travis Austin* Mark Trew†

Andrew Pullan‡

*University of Auckland, t.austin@auckland.ac.nz

†University of Auckland, m.trew@auckland.ac.nz

‡University of Auckland, a.pullan@auckland.ac.nz

Proceedings of the 2005 IEEE
Engineering in Medicine and Biology 27th Annual Conference
Shanghai, China, September 1-4, 2005

A Comparison of Multilevel Solvers for the Cardiac Bidomain Equations

Travis Austin, Mark Trew, and Andrew Pullan

Abstract— **Computing the extracellular potentials in a bidomain cardiac activation model is a computationally significant step in the solution process. Thus, using a fast solver can drastically reduce the overall time of simulation. Solving for the extracellular potentials involves inverting the matrix coming from the elliptic equation describing the extracellular-intracellular potential coupling. Elliptic equations are known to yield matrices that become progressively more ill-conditioned as the spatial resolution is increased. However, optimal multilevel solution methods are known to exist for these equations given enough effort is placed into developing the correct solution components. Two multilevel solvers that automatically perform much of this work are Black Box Multigrid (BOXMG) and Algebraic Multigrid (AMG). In this paper, we compare the performance of BOXMG and AMG as solvers for the elliptic component of the bidomain equations. Our investigation is with respect to simulations of reentry in two-dimensional cardiac tissue.**

I. INTRODUCTION

Using a robust solver to obtain extracellular potentials from the elliptic part of the discrete cardiac bidomain equations is essential to fast simulations. In [1] Vigmond *et al.* noted how this solve step can be the most time intensive part of the total simulation. Sundnes *et al.* in [2] and Weber dos Santos *et al.* in [3] used as solvers the preconditioned conjugate gradient (PCG) method with different variations of geometric multigrid (GMG) as preconditioners. Both approaches showed a significant performance gain over PCG using an incomplete LU (ILU) preconditioner. The multigrid preconditioners used in both papers, however, relied on a more expensive smoother than what is traditionally employed within GMG methods. In this paper, we introduce two multilevel methods, Black Box Multigrid (BOXMG) and Algebraic Multigrid (AMG), that pay higher costs in the setup to avoid needing expensive smoothers. Moreover, we use them directly as solvers rather than as preconditioners.

Both BOXMG and AMG are robust multilevel methods that are in some sense *black box*, meaning that a matrix and a right-hand side passed to these methods are enough to generate the components of the algorithm. What differentiates BOXMG from AMG is that BOXMG needs an underlying grid hierarchy while AMG does not. This implies that, of the two, only AMG is an applicable solver for problems defined on completely unstructured meshes. Fortunately, we concentrate here on structured meshes in two dimensions when examining the performance of the solution methods. This is not an idealized setting, though, since in practice our studies of the electrical activity of *in vitro* cardiac tissue take place on structured grids [4], [5]. As a basis for comparison to existing work, we also include results for PCG using an incomplete LU preconditioner (PCG-ILU).

The context in which this examination takes place is two-dimensional simulations of reentry [1]. All solution methods will be compared by noting setup times, solution times, memory overhead, and iteration counts. Our results show that BOXMG is nearly four orders of magnitude faster than PCG-ILU for the largest problem, while AMG is nearly three orders of magnitude faster than PCG-ILU for the same problem. We also see, though, that the setup costs for AMG are 15 times the setup costs for BOXMG and PCG-ILU. We do not examine AMG and BOXMG as preconditioners for PCG for reasons that we explain in the discussion.

Authors are with the Bioengineering Institute, University of Auckland, Auckland, New Zealand. Email: t.austin@auckland.ac.nz

II. METHODS

A. Discrete Bidomain Equations

The bidomain equations are a coupled system of nonlinear equations that model the flow of current within the intracellular and extracellular spaces of cardiac tissue. Let ϕ_i and ϕ_e be intracellular and extracellular potentials, and $\boldsymbol{\sigma}_i$ and $\boldsymbol{\sigma}_e$ be intracellular and extracellular conductivities, respectively. If the transmembrane potential is given by $V_m := \phi_i - \phi_e$ and $\mathcal{A}_i[\,\cdot\,] = -\nabla \cdot (\boldsymbol{\sigma}_i \nabla [\,\cdot\,])$ and $\mathcal{A}_e[\,\cdot\,] = -\nabla \cdot (\boldsymbol{\sigma}_e \nabla [\,\cdot\,])$, we can write the bidomain equations as [6]

$$A_m C_m \frac{\partial V_m}{\partial t} + \mathcal{A}_i V_m = -\mathcal{A}_i \phi_e - A_m I_{ion}(V_m) \quad (1)$$

$$(\mathcal{A}_i + \mathcal{A}_e)\phi_e = -\mathcal{A}_i V_m + i_e(t). \quad (2)$$

Here, A_m is the surface-to-volume ratio of the cell membrane and C_m is the membrane capacitance per unit area. I_{ion} is the sum of all ionic currents, which depends on the choice of cellular electrical activity model. The model that we employ in this work is the Beeler-Reuter Drouhard-Roberge (BRDR) ionic model [7] modified to allow for large transmembrane potentials. Other parameter values were taken from [8]. Lastly, i_e is used to allow current injections per unit volume into the extracellular space.

Specifying the boundary conditions for (1) and (2) is necessary to create a well-posed problem. The conventional assumption is there is no current flow from the intracellular domain to the extramyocardial domain. This leads to

$$(\boldsymbol{\sigma}_i \nabla \phi_i) \cdot \mathbf{n} = 0 \quad (3)$$

implying

$$(\boldsymbol{\sigma}_i \nabla V_m) \cdot \mathbf{n} = -(\boldsymbol{\sigma}_i \nabla \phi_e) \cdot \mathbf{n} \quad (4)$$

are the boundary conditions on V_m. In contrast to the intracellular domain, current may flow from extracellular to extramyocardial. Furthermore, current is conserved, i.e.,

$$(\boldsymbol{\sigma}_e \nabla \phi_e) \cdot \mathbf{n} = (\boldsymbol{\sigma}_o \nabla \phi_o) \cdot \mathbf{n}, \tag{5}$$

given $\boldsymbol{\sigma}_o$ and ϕ_o are extramyocardial conductivity and potential. Experiments arise where the extramyocardial is a bath solution or a "torso". When it is neither, then the right-hand side of (5) equals zero.

We use a semi-implicit split-step strategy for time integration ([8]). This utilizes a first-order forward step in time to approximate $\partial V_m/\partial t$. A finite element spatial discretization of domain $\Omega = [0,a]\times[0,b]$ using rectangular tensor-product finite elements leads to the following algorithm used in updating V_m and ϕ_e at each time step.

1. Calculate $\mathbf{I}_{ion}^{t+\Delta t/2}$ from $\mathbf{V}^t$ using the BRDR ionic model.
2. Given $\mathbf{V}^t$, $\mathbf{\Phi}^t$, and $\mathbf{I}_{ion}^{t+\Delta t/2}$, solve
$$(C_m\mathbf{M} + \Delta t\, \mathbf{A}_i)\mathbf{V}^{t+\Delta t} = C_m\mathbf{M}\mathbf{V}^t - \Delta t\, \mathbf{A}_i\mathbf{\Phi}^t - \mathbf{M}\,\mathbf{I}_{ion}^{t+\Delta t/2} \tag{6}$$
for $\mathbf{V}^{t+\Delta t}$.
3. Given $\mathbf{V}^{t+\Delta t}$ and source $\mathbf{I}_e^{t+\Delta t}$, solve
$$(\mathbf{A}_i + \mathbf{A}_e)\mathbf{\Phi}^{t+\Delta t} = -\mathbf{A}_i\mathbf{V}^{t+\Delta t} + \mathbf{I}_e^{t+\Delta t} \tag{7}$$
for $\mathbf{\Phi}_e^{t+\Delta t}$.

Step 3 corresponds to the elliptic solve that for many simulations can consume the majority of the total simulation time. Thus, it is the matrix $(\mathbf{A}_i+\mathbf{A}_e)$ that we need to approximately invert with a multilevel method to get extracellular potentials at the current time step. Next, we describe the two multilevel methods.

B. BOXMG and AMG

Both BOXMG and AMG belong to the family of multilevel methods that depend on the same multilevel framework: smoothing $\rightarrow$ restriction $\rightarrow \cdots$ coarse-grid correction $\rightarrow \cdots$ interpolation $\rightarrow$ smoothing [9]. What makes the multilevel solution algorithm tick is that high frequency error components can be eliminated by fine grid smoothings and low frequency error components by coarse-grid corrections.

Most multigrid users utilize the classical geometric multigrid method that relies on standard n-linear interpolation. BOXMG and AMG do not use such interpolation, but instead use matrix-dependent interpolation. Proper matrix-dependent interpolation normally leads to better convergence, in particular, in the case of discontinuous data. Additionally, both BOXMG and AMG define coarse-grid operators using a Galerkin approach. The difference between BOXMG and AMG is what information each method takes advantage of in setting up the collection of multilevel components. BOXMG assumes that there is a given grid hierarchy (i.e., a structured problem), while AMG makes no assumptions other than there is a matrix and a right-hand side. We briefly define each method below.

To describe BOXMG, we need a hierarchy of grids. Thus, assume that spatial discretization of Ω leads to a collection of finite element nodes given by

$$\Omega_K := \{\,(x_i, y_j) \;:\; i = 0, 2^m \text{ and } j = 0, 2^n\}, \tag{8}$$

where

$$0 = x_0 \le x_1 \le \cdots \le x_{2^m} = a$$

and

$$0 = y_0 \le y_1 \le \cdots \le y_{2^n} = b.$$

Here, m and n are given positive integers. A hierarchy of grids can be defined as $\Omega_K \supset \Omega_{K-1} \supset \cdots \Omega_0$. We get Ω_{K-1} from Ω_K by removing (x_i, y_j) from Ω_K if i or j is odd. Subsequent coarser grids are obtained in a similar manner. Moreover, each Ω_ℓ can be expressed as (8) for a given m and n.

Next, we define

$$\Omega_\ell^C := \{(x_i, y_j) \;:\; i \text{ is even, } j \text{ is even}\}$$

and $\Omega_\ell^F := \Omega_\ell \backslash \Omega_\ell^C$ as the collection of coarse grid points and fine grid points, respectively. If we let $\mathbf{A}_\ell$ denote the matrix coming from a finite element discretization on Ω_ℓ, then we can write

$$\mathbf{A}_\ell := \begin{pmatrix} \mathbf{A}_F & \mathbf{A}_{FC} \\ \mathbf{A}_{CF} & \mathbf{A}_C \end{pmatrix}, \tag{9}$$

where the two rows correspond to the separate fine and coarse grid functions. With BOXMG we generate an interpolation operator $\mathbf{P}_{\ell-1}^\ell$ that maps functions defined grid $\Omega_{\ell-1}$ to functions defined on grid Ω_ℓ. The matrix-dependent interpolation operator is of the form

$$\mathbf{P}_{\ell-1}^\ell = \begin{pmatrix} -\widetilde{\mathbf{A}}_F^{-1}\widetilde{\mathbf{A}}_{FC} \\ I \end{pmatrix}$$

so that

$$\mathbf{P}_{\ell-1}^\ell v_{\ell-1} = (v_\ell^F, v_\ell^C)^T$$

for $v_{\ell-1}$ defined on $\Omega_{\ell-1}$ and v_ℓ^F and v_ℓ^C defined on Ω_ℓ^F and Ω_ℓ^C, respectively. BOXMG then uses grid and the preservation of flux continuity to define approximations $\widetilde{\mathbf{A}}_F$ and $\widetilde{\mathbf{A}}_{FC}$. See [10] or [11] for details. Lastly, we note that the Galerkin approach to defining coarse-grid equations implies

$$\mathbf{A}_{\ell-1} = (\mathbf{P}_{\ell-1}^\ell)^T \mathbf{A}_\ell \mathbf{P}_{\ell-1}^\ell. \tag{10}$$

We just saw how an underlying grid structure is needed by BOXMG. Next, we turn to AMG which needs no grid structure. Of course a price is paid for this by AMG in that setup is more intensive. Be that as it may, setup is the heart of Algebraic Multigrid. The setup produces a proper subset of coarse grid points, assuming no grid, and an accurate interpolation operator. It is impossible in this short space to get the reader to fully appreciate the details of the setup. See [12] if interested. That said, we will describe the two defining features of AMG that are accounted for in the setup: the notion of algebraic smoothness and strong dependencies.

We begin with the idea that AMG fixes the smoother to be Gauss-Seidel and from this defines what is *algebraically*

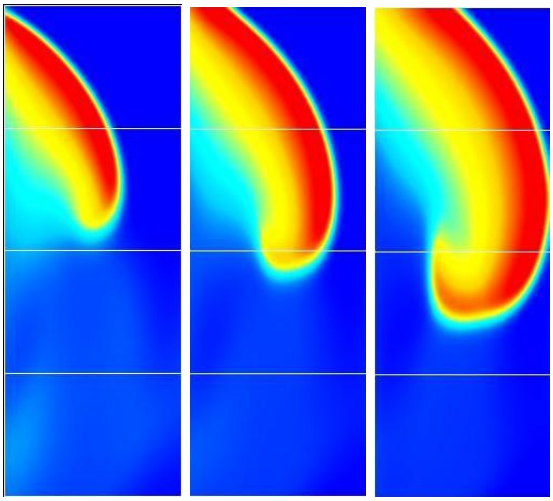

Fig. 1. Transmembrane Potential at t = 95 ms, 100 ms, 105 ms

smooth error. To describe this notion, let $\mathbf{A}$ denote the matrix and $\mathbf{b}$ denote the right-hand side. Additionally, given $\widetilde{\mathbf{x}}$ is an approximation to true solution $\mathbf{x}$, we put $\mathbf{r} = \mathbf{b} - \mathbf{A}\widetilde{\mathbf{x}}$ to be the residual and $\mathbf{e} = \mathbf{x} - \widetilde{\mathbf{x}}$ to be the error. We then get the standard residual equation, $\mathbf{Ae} = \mathbf{r}$.

Thus, we say that an error is *algebraically smooth* if

$$\mathbf{Ae} \approx \mathbf{0}, \tag{11}$$

which is equivalent to $\mathbf{r} \approx \mathbf{0}$. Examining each row defined by (11) implies

$$a_{ii}e_i \approx -\sum_{j \neq i} a_{ij}e_j, \tag{12}$$

where $\mathbf{e} = \{e_i\}$ and $\mathbf{A} = \{a_{ij}\}$. What (12) says is that e_i can be accurately approximated by a weighted average of its neighbors. This is not always the same as geometrically smooth. See [9] for examples.

We use (12) again to define the concept of *strong dependence*. Strong dependencies are needed to define a "good" set of coarse grid points that will accurately interpolate to our fine grid point. For unknown x_i, we have an equation of the same form as (12). To pick a possible set of coarse grid points to use in interpolating to x_i, we determine from (12) which other unknowns are important in determining x_i. These are given by the x_j that have a large relative a_{ij}. In AMG terminology, we say x_i *strongly depends* on x_j. It is this kind of approach that underlies the selection of the coarse grid at each level.

AMG uses the notion of *strong dependence* and *algebraic smoothness* to define both the set of coarse grid points and the interpolation operator. The end result is a collection of coarse grid points on which smooth components can be represented accurately and an interpolation operator that accurately transfer information from the coarse grid to the fine grid. The remaining piece of AMG is that the coarse-grid operators are defined via a Galerkin approach, exactly the same as in (10).

TABLE I
SETUP TIMES FOR AMG, BOXMG, AND PCG-ILU, AND A MATVEC.

# GRID POINTS	AMG	BOXMG	PCG-ILU	MATVEC
174947	1.70 s	0.13 s	0.11 s	0.009 s
331331	3.88 s	0.25 s	0.23 s	0.015 s
752001	8.37 s	0.53 s	0.51 s	0.034 s

C. Numerical Results

The test problem used to compare BOXMG to AMG was defined on a 5 mm $\times$ 14 mm 2D slice of cardiac tissue. The extracellular potential had zero Dirichlet boundary conditions at $x = 0$ mm and $x = 5$ mm and zero Neumann boundary conditions at $y = 0$ mm and $y = 14$ mm. We started the simulation at $t = 0$ ms with a plane wave from $x = 0$ to $x = 5$. At 42 ms, a part of the lower left quartile was stimulated in order to initiate the spiral wave that is characteristic of reentry. For 100 ms, we allowed the spiral wave to propagate through the tissue before ending the simulation. Images of the spiral wave at $t = 95$ ms, 100 ms, and 105 ms are illustrated in Figure 1. We ran the simulations on three grids having 174947, 331331, and 752001 dofs. All computations were performed on a 32-bit 2.8 Mhz Pentium 4 machine with 2 Gb physical RAM and 512 Kb L2 cache.

In the simulations, the tolerance (stop_tol) for the elliptic solve was set to be 10^{-6}. We chose as our stopping criteria one that compares the current residual norm (using a standard discrete least-squares norm) to the corresponding norm of the right-hand side [13]. That is, for current estimate $\mathbf{x}_k$, we defined convergence to be whenever

$$\|\mathbf{r}_k\|_2 \leq \text{stop\_tol} \cdot \|\mathbf{b}\|_2, \tag{13}$$

given that $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$. Additionally, if $\|\mathbf{b}\|_2$ was less than 10^{-10} upon entering a solver, the solution method was immediately terminated and the solution was set to zero.

Setup times are given in Table I. The last column is the time to perform one matrix-vector (matvec) computation. We deduce from this that the AMG setup costs approximately 246 matvecs, while the BOXMG and PCG-ILU setup cost approximately 15. The first set of timings for the solve phase that we include are for a 15 ms window of time from $t = 35$ ms to $t = 50$ ms. A time step of 0.01 ms was taken requiring 1500 time steps (and elliptic solves). We compare all three solvers in this time window. The significance of this window is that the spiral wave is being initiated at $t = 42$ ms. See Table II for these results. Note that we chose not to compare all three solvers for the for 100 ms because of the extensive time required by PCG-ILU.

We performed full simulation runs (100 ms) with AMG and BOXMG and measured the subsequent times. For the grid with 331331 dofs, the simulation using BOXMG as the solver required approximately *11 hrs and 6 minutes*. AMG used as the solver for the same simulation took *12 hrs and 20*

TABLE II
SOLVE TIMES (**Iteration Counts**) FOR AMG, BOXMG, AND PCG-ILU FOR 15 MS WINDOW.

# NODES	AMG	BOXMG	PCG-ILU
174947	1647 s (**3516**)	1556 s (**3738**)	3091 s (**33508**)
331331	3300 s (**3806**)	2993 s (**3706**)	8022 s (**45769**)
752001	11164 s (**5992**)	7308 s (**4468**)	30998 s (**96119**)

minutes. The number of total iterations required by BOXMG and AMG were 40998 and 45500 respectively.

III. CONCLUSION

In Table II, we see that the two multilevel solvers clearly generate faster simulations. Furthermore, it appears the growth in the number of iterations required by the two multilevel solvers is not as substantial as for PCG-ILU. If we were to run these simulations at even higher degrees of resolution, we in fact should see the number of iterations required by the two multilevel solvers approach an asymptotic limit. This would not be the case for PCG-ILU.

The cost of the two multilevel solvers, besides the setup, appears with the need for more memory. BOXMG needs approximately twice as much memory as PCG-ILU to solve the same problem. This memory is used for setting up all of the coarse grid components. Similar components are set up by AMG but at a higher memory overhead. AMG requires approximately four times the memory of PCG-ILU.

As mentioned in the introduction, we have been seeing some usage of multilevel-based solvers (as preconditioners for PCG) in the context of the bidomain equations [2], [3]. Here, we have chosen to use the multilevel solvers not as preconditioners but directly as the solvers. There are two reasons that we have made that choice. The first is that on smaller problems we noticed no advantage to using PCG with the two multilevel solvers as preconditioners: the multilevel solvers alone were just as fast. The second is a desire to reduce computations on the finest grid.

As can be seen from Figure 1, successive time steps (several time steps in the figure) of the bidomain equations only lead to, in many cases, local changes to the solution from the previous time step. Thus, the solution to the previous time step often is a very accurate approximation to the current time step. This has been noted in several papers (see [1], [2], [3]). Thus, one should be able to use an adaptive solution algorithm whereby only smoothings occur near where the solution is active. In our example, this would be near where the spiral wave is propagating.

If a PCG method was used with a multilevel preconditioner, then we would be required to perform many matrix-vector computations on the finest grid. Instead, if we used BOXMG or AMG as the solver, then we could focus the computation where it most mattered and avoid unneeded computations on the grid of highest resolution. Currently, we are investigating this type of adaptive algorithm for both AMG and BOXMG.

Robust multilevel solvers like BOXMG and AMG have yet to be considered by the bioengineering community for solving the bidomain equations. Here, we have introduced both solvers and compared their performance relative to PCG-ILU. In [14], we explicitly compare the performance of BOXMG to GMG, both as solvers and as preconditioners. We also investigate the effect felt by the multilevel solvers from discontinuities in the electrical conductivities.

ACKNOWLEDGMENTS

The authors acknowledge the support of the Health Research Council of New Zealand. Also, thanks are due to John Ruge, Klaus Stüben, and Rolf Hempel and the Fraunhöfer Institute of Algorithms and Scientific Computing (SCAI) for allowing the usage of AMG1R6, and to Joel Dendy and Dave Moulton and Los Alamos National Laboratory for the usage of BOXMG.

REFERENCES

[1] E. Vigmond, F. Aguel, and N. Trayanova, "Computational techniques for solving the bidomain equations in three dimensions," *IEEE Trans. Bio-Med. Eng.*, vol. 49, no. 11, pp. 1260–1269, 2002.

[2] J. Sundnes, G. Lines, K.-A. Mardal, and A. Tveito, "Multigrid block preconditioning for a coupled system of partial differential equations modeling the electrical activity of the heart," *Computer Methods in Bio-Mech and Bio-Med Eng.*, vol. 5, no. 6, pp. 397–409, 2002.

[3] R. Weber dos Santos, G. Plank, S. Bauer, and E. Vigmond, "Parallel multigrid preconditioner for the cardiac bidomain model," *IEEE Trans. Bio-Med. Eng.*, vol. 51, no. 11, pp. 1960–1968, Nov. 2004.

[4] D. Hooks, K. Tomlinson, S. Marsden, L. I., B. Smaill, and A. Pullan, "Cardiac microstructure: Implications for electrical propagation and defibrillation in the heart," *Circulation Research*, vol. 9, no. 4, pp. 331–338, 2002.

[5] M. Trew, I. LeGrice, B. Smaill, and A. Pullan, "A finite volume method for modeling discontinuous electrical activation in cardiac tissue," *Annals of Bio-Med Eng.*, vol. 33, no. 5, 2005.

[6] J. Keener and J. Sneyd, *Mathematical Physiology*. New York: Springer-Verlag, 1998.

[7] K. Skouibine, N. Trayanova, and P. Moore, "A numerically efficient model for simulation of defibrillation in an active bidomain sheet of myocardium," *Mathematical Biosciences*, vol. 166, pp. 85–100, 2000.

[8] M. Buist, G. Sands, P. Hunter, and A. Pullan, "A deformable finite element derived finite difference method for cardiac activation problems," *Annals of Bio-Med Eng.*, vol. 31, pp. 577–588, 2003.

[9] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial: Second Edition*, 2nd ed. Pennsylvania, PA: SIAM, 2001.

[10] J. Moulton, J. Dendy Jr., and J. Hyman, "The black box multigrid numerical homogenization algorithm," *J. of Comp. Phys.*, vol. 142, pp. 80–108, 1998.

[11] S. Knapek, "Matrix-dependent multigrid homogenization for diffusion problems," *SIAM J. Sci. Computing*, vol. 20, no. 2, pp. 515—533, 1998.

[12] J. Ruge and K. Stüben, *Multigrid Methods*. Philadelphia, Penn.: SIAM, 1987, ch. Algebraic multigrid, pp. 73–130.

[13] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[14] T. Austin, M. Trew, and A. Pullan, "Solving the cardiac bidomain equations for discontinuous conductivities," in preparation.