Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006



# Accelerating Large Cardiac Bidomain Simulations by Arnoldi Preconditioning

Makarand Deo*, Steffen Bauer†, Gernot Plank‡, and Edward Vigmond*
*University of Calgary, Calgary, Canada. Email: mdeo@ucalgary.ca
†Physikalisch-Technische Bundesanstalt, Berlin, Germany.
‡Medical University of Graz, Institute of Biophysics, Graz, Austria.

*Abstract*— **Bidomain simulations of cardiac systems often involve solving large, sparse, linear systems of the form Ax=b. These simulations are computationally very expensive in terms of run time and memory requirements. Therefore, efficient solvers are essential to keep simulations tractable. In this paper, an efficient preconditioner for the conjugate gradient (CG) method based on system order reduction using the Arnoldi method (A-PCG) is explained. Large order systems generated during cardiac bidomain simulations using a finite element method formulation, are solved using the A-PCG method. Its performance is compared with incomplete LU (ILU) preconditioning. Results indicate that the A-PCG estimates an approximate solution considerably faster than the ILU, often within a single iteration. To reduce the computational demands in terms of memory and run time, the use of a cascaded preconditioner is suggested. The A-PCG can be applied to quickly obtain an approximate solution, subsequently a cheap iterative method such as successive overrelaxation (SOR) is applied to further refine the solution to arrive at a desired accuracy. The memory requirements are less than direct LU but more than ILU method. The proposed scheme is shown to yield significant speedups when solving time evolving systems.**

## I. Introduction

The numerical solution of partial differential equations (PDE) associated with bidomain problems often lead to large linear systems of the form $\mathbf{A}x = b$. Being an initial value problem, the system requires the solution of the matrix equation at every time step in order to update the system. This consumes the most resources, both in terms of computation and memory requirements.

Most bidomain solution schemes implemented so far have applied an operator splitting technique to decouple the system, yielding a linear elliptic PDE, a linear parabolic PDE, and a set of nonlinear ODEs [1]. The solution of the elliptic portion typically dominates the computation time [1]. Different methods such as the finite element method [2], the finite volume method [3], or the finite difference method [4] have been applied to facilitate spatial and temporal discretization.

This paper examines the use of an Arnoldi preconditioner to speed up the solution of large, sparse linear systems. A finite element formulation of the decoupled bidomain equations is generated for systems of orders of up to half a million. The resulting linear system is solved at each time step to study the electrical propagation using the CG solver with two different preconditioning methods: Arnoldi (A-PCG), and Incomplete LU (ILU). The preconditioner performance is evaluated by comparing setup times, memory usage, solution times and the number of iterations required to converge. An improved modification to the A-PCG which aims at solving large cardiac systems accurately using reduced computational resources by applying successive overrelaxation (SOR) refinement step was also implemented.

## II. Methods

### A. Governing equations

Cardiac electrical activity is described by the bidomain equations which account for current flow in both the intracellular and the extracellular domain. The potential fields, $\phi_i$ and $\phi_e$ in the respective domains, are linked through the transmembrane current density, $I_m$. Decoupling the equations results in an elliptic and a parabolic equation with $\phi_e$ and the transmembrane voltage, $V_m$ ($= \phi_i - \phi_e$) as the independent variables [5]:

$$\nabla \cdot (\overline{\sigma}_i + \overline{\sigma}_e)\nabla\phi_e = -\nabla \cdot \overline{\sigma}_i \nabla V_m - I_e \tag{1}$$

$$\nabla \cdot \overline{\sigma}_i \nabla V_m = -\nabla \cdot \overline{\sigma}_i \nabla\phi_e + \beta I_m \tag{2}$$

where $\overline{\sigma}_i$ and $\overline{\sigma}_e$ are the intracellular and extracellular conductivity tensors, respectively, $I_e$ is an extracellular current density stimulus, $\beta$ is the surface to volume ratio of the cardiac cells and $I_m$ is the membrane current density. Equations (1) and (2) are solved sequentially as an elliptic problem and a parabolic problem. Typically, it is the elliptic equation (Eqn. 1) which dominates computation in bidomain simulations [6]. Using the finite element method (FEM), 3D tissue slabs of varying size (between $1 \times 1 \times 0.05\ cm^3$ and $3 \times 3 \times 0.05\ cm^3$) were disretized at a resolution of 100 $\mu m$ resulting in system sizes between $N = 50500$ and $N = 451500$. The resulting disretized form of Eqn. 1 is,

$$\mathbf{K_{i+e}}\boldsymbol{\phi}_\mathbf{e}^\mathbf{t} = -\mathbf{K_i}\mathbf{v}_\mathbf{m}^\mathbf{t} - \mathbf{M}\mathbf{I}_\mathbf{e}^\mathbf{t} \tag{3}$$

where $\mathbf{K}$ is the FEM stiffness matrix and $\mathbf{M}$ is the FEM lumped mass matrix. Both matrices were computed using linear tetrahedral elements with the subscript on $\mathbf{K}$ denoting whether the matrix was created using $\overline{\sigma}_i$, or the sum of the conductivities, $\overline{\sigma}_i + \overline{\sigma}_e$. The dynamic membrane behaviour was described by the modified Beeler-Reuter-Drouhard-Roberge model [7].

1-4244-0033-3/06/$20.00 ©2006 IEEE.

### B. Arnoldi algorithm for (partial) reduction

Model order reduction techniques are based on approximating the original system matrix $\mathbf{A}$ with a lower order matrix $\hat{\mathbf{A}}$ such that the eigenvalues of $\hat{\mathbf{A}}$ are reasonable approximations of the leading eigenvalues of $\mathbf{A}$ [8].

One popular approach used for partial reduction of $\mathbf{A}$ to a smaller upper Hessenberg matrix $\mathbf{H}$ using an incompletely computed orthogonal matrix $\mathbf{Q}$ is the Arnoldi algorithm [8]. Assume $\mathbf{Q} = [\mathbf{q}_1\ \mathbf{q}_2\ ...\ \mathbf{q}_m]$, where $\mathbf{q}_i$ represents the $i^{th}$ column of matrix $\mathbf{Q}$. A similarity transformation $\mathbf{AQ} = \mathbf{QH}$ is used, which can be written as

$$\mathbf{A}\begin{bmatrix} [\mathbf{q}_1] & [\mathbf{q}_2] & ... & [\mathbf{q}_m] \end{bmatrix} = \begin{bmatrix} [\mathbf{q}_1] & [\mathbf{q}_2] & ... & [\mathbf{q}_m] \end{bmatrix} \times \begin{bmatrix} h_{11} & h_{12} & h_{13} & ... \\ h_{21} & h_{22} & h_{23} & ... \\ 0 & h_{32} & h_{33} & ... \\ ... & ... & ... & ... \end{bmatrix}$$

The above equation is solved by equating the expressions for each row on both sides with the condition that the columns are orthonormal ($\mathbf{q}_i^T\mathbf{q}_j = \delta_{ij}$).

Thus only a few leading eigenvalues of $\mathbf{A}$ are extracted through the incomplete similarity transformation $\mathbf{AQ} = \mathbf{QH}$, resulting in a low order matrix $\mathbf{H}$ which is an approximation of $\mathbf{A}$.

*a) Choice of starting vectors:* The choice of starting vectors is crucial because the Arnoldi algorithm extracts eigenvalues of $\mathbf{A}$ in the space spanned by the columns of $\mathbf{Q}$. In this study, a unity norm vector in the direction of $\mathbf{b}$ is chosen as a starting vector.

*b) Orthogonalization:* In a Krylov space, the column vectors of $\mathbf{Q}$ are repetitively multiplied by $\mathbf{A}$. As a result of this multiplication, the columns of the space tend towards the dominant eigenvectors of $\mathbf{A}$, and the difference between two successive columns becomes very small. To maintain accuracy, the columns have to be orthogonalized during their generation. Hence, before a new column is generated, the most recent vector (column) is orthogonalized with respect to all previous vectors (columns) [9]. In practice, however, it is observed that columns in the Krylov space which were orthogonalized still had non-zero inner products of the order $10^{-9}$. In such cases, a reorthogonalization procedure has to be applied when more eigenvalues need to be extracted.

## III. Use of Arnoldi matrix as a preconditioner

Biomedical problems, because of their bigger size and complexity, require considerably higher orders of extraction, $m$, which can be a limitation where memory and/or time is an issue. The algorithm extracts eigenvalues of the system matrix in the space spanned by the vectors of $\mathbf{Q}$. Therefore, the Arnoldi method does not give accurate solutions with smaller $m$, when used as a solver. This section discusses an alternative use of the Arnoldi matrix as a preconditioner for the CG method. This approach reduces the memory constraints by opting for lower values of $m$. The reduced order matrix $\mathbf{H}$ obtained by the Arnoldi method is used to determine the conjugate directions along the steepest descent of the residual.

### A. The A-PCG algorithm

In each iteration of PCG, the new direction search vector, $\mathbf{p}$, is determined by solving

$$\mathbf{Mp} = \mathbf{r} \tag{4}$$

where $\mathbf{M}$ is the preconditioning matrix, and $\mathbf{r}$ is the residual vector [10] calculated as

$$\mathbf{r} = \mathbf{b} - \mathbf{Ax} \tag{5}$$

We propose a modification to Eqn. 4 by mapping the residual vector $\mathbf{r}$ into a reduced vector $\widehat{\mathbf{r}}$,

$$\widehat{\mathbf{r}} = \mathbf{Q}^T\mathbf{r} \tag{6}$$

and then solving

$$\mathbf{H}\widehat{\mathbf{p}} = \widehat{\mathbf{r}} \tag{7}$$

to obtain the reduced direction vector $\widehat{\mathbf{p}}$, which is then mapped back to the original dimension direction vector $\mathbf{p}$ using $\mathbf{Q}$,

$$\mathbf{p} = \mathbf{Q}\widehat{\mathbf{p}} \tag{8}$$

The reduced matrix, $\mathbf{H}$, is calculated using the Arnoldi method as explained above. This modification facilitates the use of any square matrix as a preconditioner in PCG method, irrespective of its size.

### B. A-PCG with SOR refinement (A-PCG/SOR)

While solving large systems, such as those generated during cardiac simulations, the choice of $m$ needed to maintain the required accuracy increases which increases the setup time as well as memory usage for the Arnoldi computation. In order to optimize the number of A-PCG iterations and setup time, use of cascaded preconditioners is proposed. In this case, a relatively smaller $m$ is used for the A-PCG iteration to obtain an approximate solution which is then refined using a few iterations of SOR method. The residual vector norm is rapidly decreased during the A-PCG iteration while the SOR iterations help to finetune the solution to the required accuracy. This approach significantly cuts down the preconditioner setup time as well as memory consumption.

### C. Tuning the A-PCG/SOR method

To determine the optimal choice of $m$, 5 $ms$ of action potential propagation were simulated in a 3D tissue slab (N=201000) by applying a transmembrane current stimulus to the upper left corner of the tissue. The system was solved with Arnoldi preconditioners of different sizes obtained by increasing $m$ from 10 to 170 in steps of 10. The optimal $m$ was chosen to be as small as possible, to reduce the computation time, while preserving the accuracy as measured by the $L_\infty$ norm of the residual.

### D. Comparison with other Preconditioners

The proposed A-PCG/SOR was evaluated by comparing run times and number of iterations with an incomplete LU preconditioner (ILU) which preserved the sparsity pattern of $\mathbf{A}$. The same activation sequence as used for determining the optimal $m$ was repeated for all tissue sizes under study.

As a further test, the initiation of a single rotor reentry by a S1-S2 cross shock protocol was chosen as a test sequence [1] 500 ms of reentrant activity were simulated. In this case, a 2D tissue sheet ($4 \times 2\ cm^2$) was used resulting in a system size of $N$ =80000.

For A-PCG, a convergence criterion based on the norm of the preconditioned residual vector was set as $\|\mathbf{r}_i\| \leq \varepsilon \|\mathbf{r}_0\|$, where $\mathbf{r}_i$ is the residual vector in $i^{th}$ iteration and $\varepsilon = 1 \times 10^{-9}$ is the error tolerance [11].

All code was written in Matlab (Mathworks Inc., Boston). Timing information is given for uncompiled Matlab code executed on a 2 GHz PC running Linux.

## IV. Results

### A. Selection of the Appropriate Order of Extraction

In Fig. 1, the $L_\infty$ of the residual (averaged over 50 time steps) (in $\mu V$) is shown as a function of $m$ for $N$ =201000. With increasing $m$, the residual error decreased which implies that a more accurate solution was obtained. This significant reduction in the residual is obtained using just a single A-PCG iteration in each case. However, increasing $m$ also increases the run time as well as memory requirements significantly. During the simulations, the $L_\infty$ norm was further reduced to $5 \times 10^{-2} \mu V$ using a few SOR refinement iterations ($< 40$). To make an appropriate choice of $m$ for a given system, a trade-off between $L_\infty$ of the residual and the total time required to converge has to be made. For $N$ =201000, choosing $m$ =150 with an $L_\infty$ error tolerance of $5 \times 10^{-2} \mu V$ required a single A-PCG and 39 SOR iterations and the convergence time was 66.16 seconds/time step. Setting $m$ =170 decreased the SOR iterations to 35 but increased the computation time to 75.08 seconds/time step. The optimal $m$ was determined to be 150 based on the criterion that the $L_\infty$ error should be less than $5 \times 10^{-2} \mu V$ while keeping the computation time approximately less than 25% of that of ILU. In this paper, for all further runs, if not otherwise noted, $m$ was set to be 150.

### B. Performance Comparison between Preconditioners

Based on the number of iterations required and the CPU run time, Table I summarizes performance results obtained with A-PCG/SOR and ILU [10] preconditioners. Regardless of $N$, the A-PCG led to the approximate solution within a single iteration which was further refined by SOR. The ILU preconditioning required more iterations as $N$ was increased, with an approximate three-fold increase for ILU as $N$ increased from 50500 to 451500. The number of SOR iterations required for refining the solution, however, did not show a considerable increase. For the largest system considered ($N = 451500$), the A-PCG/SOR method required $\approx$ 11 times less iterations than the ILU method.

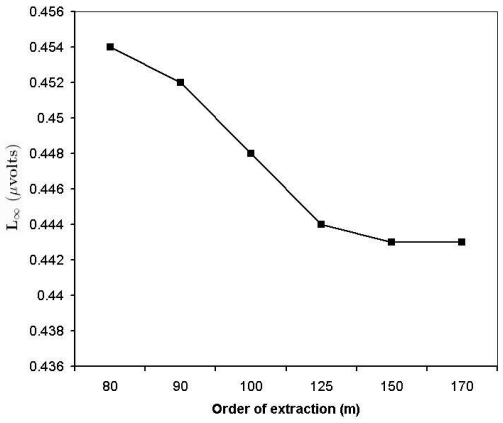


Fig. 1. Reduction in the $L_\infty$ norm of the residual (in $\mu V$)with increasing orders of Arnoldi extraction ($m$) for a system of original order 201000. The optimal choice of $m$ was found to be 150, considering the trade-off between accuracy and time.

TABLE I

Iterative Solver Performance. The number of iterations, CPU run time and CPU setup time are given for Conjugate Gradient method with incomplete LU (ILU) and Arnoldi preconditioning with SOR refinement (A-PCG/SOR) Note: The timings for A-PCG/SOR given below include both the A-PCG and SOR.

| System order | Number of Iterations | | Run Time (s) | |
|---|---|---|---|---|
| | ILU | A-PCG/SOR | ILU | A-PCG/SOR |
| 50500 | 188 | 1/37 | 45.89 | 15.45 |
| 111589 | 301 | 1/40 | 164.22 | 30.57 |
| 201000 | 373 | 1/39 | 390.71 | 66.16 |
| 451500 | 546 | 1/50 | 1360.12 | 123.19 |

The time to converge increased approximately eight times using A-PCG/SOR as $N$ was increased from 50500 to 451500. For ILU, the time increased from 45.89 to 1360.12 seconds which corresponds to a factor of 29.6. For the largest $N$, a single time iteration with ILU preconditioning took 1360.12 seconds while a single time iteration with Arnoldi preconditioning took 123.19 seconds, which is a reduction by 91%.

### C. Simulation of reentrant activity

After validating the A-PCG/SOR method, it was tested for longer cardiac simulations involving a changing solution vector because eigenvalues are extracted based on a perticular solution. Spiral wave reentry was initiated in a 2D piece of tissue discretized with 80000 nodes and the electrical activity was simulated for 500 ms.

Fig. 2 shows extracellular potential maps obtained at various time points during the simulation. The A-PCG/SOR method reproduced the results very accurately and efficiently which were compared to ILU for accuracy. After every 25 timesteps, 10 Arnoldi vectors were recomputed and augmented to the previous transformation to account for the updates in the constantly varying solution vector.

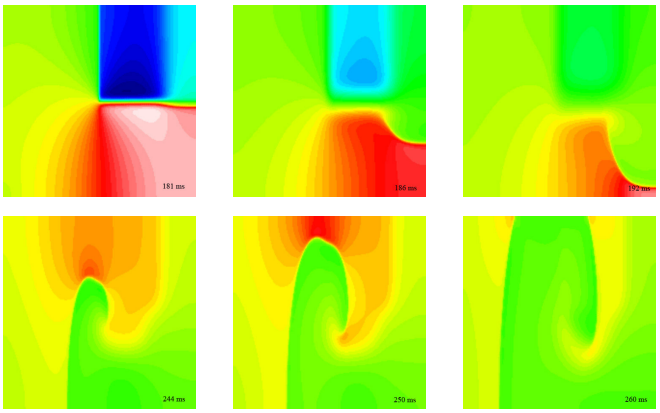

Fig. 2. Generation of a spiral wave reentry in a 2D piece of cardiac tissue. The solution at each time step is obtained using the A-PCG/SOR method. The blue corresponds to $-50\ mV$ and the red corresponds to $+25\ mV$.

## V. Discussion

In this study, we have demonstrated that the Arnoldi system reduction method can be used as an efficient preconditioner for the CG method. The proposed A-PCG approaches the true solution vector in typically one iteration, much fewer than any other preconditioner. The SOR refinement scheme further adds to the flexibility of the A-PCG method. The run time as well as setup times can be reduced significantly by choosing a lower $m$ at the cost of slightly increased number of refinement iterations. The SOR was chosen for simplicity, but use of other sophisticated preconditioners is possible which may be faster. The choice of the refinement conditioner is based on memory available and ease of implementation.

If the A-PCG method is to be used in a parallel computing environment, the computation of $\mathbf{Q}$ could be split across processors and multiplication would parallelize fairly well since the matrices are dense. The reduced system is small enough to be solved on a single node since $\mathbf{H}$ and its LU decomposition each fit into less than a megabyte of memory for orders of extraction up to $m$ =300.

The most efficient preconditioner for the elliptic portion of the bidomain equations to date has used a geometric multigrid method (GMG) [6]. Running on a single CPU, the GMG preconditioner was found to be about three times faster than ILU, but consumed 40% more memory. Results obtained in this study suggest that the A-PCG preconditioner is potentially faster than a GMG preconditioner. Assuming a ratio of three between GMG and ILU suggests that A-PCG could be up to 4 times faster than GMG. Further, implementing the A-PCG method is simpler than the GMG since the need to generate coarser grids is obviated as well as the prolongation and restriction of values between grids. Geometric variants of multigrid methods are particularly difficult to implement when domains are discretized on unstructured grids. For the A-PCG method, however, the structure of the underlying grid is irrelevant. The memory usage of the A-PCG is considerably higher, which can limit its use for very large problems when memory is an issue. But the proposed modification of SOR refinement solves this problem by opting for significantly lower values of $m$. The A-PCG is certainly quicker for a single system solution than direct methods, and for time evolving systems has outperformed ILU. Finally, if the eigenvalues associated with the system solution do not change appreciably over several time steps, it is possible to reuse the same transformation, and thereby, gain additional savings in computation time. A more thorough study is required to exploit this possibility.

## VI. Conclusions

In this study, A-PCG preconditioner for the CG method is shown to accelerate convergence when solving large cardiac systems. The A-PCG method can handle any low order preconditioner irrespective of its dimensions by mapping the direction vectors of the CG solver to a corresponding reduced subspace. Benchmark results suggest that using the Arnoldi method to obtain a reduced order preconditioner is well suited to accelerate convergence. Various high order bidomain systems of up to half million unknowns are solved, requiring just a single iteration of A-PCG. An efficient modification of A-PCG with SOR refinement adds flexibility to the algorithm by finetuning the solution for arbitrary accuracy. This technique has the potential to accelerate biomedical system simulations involving repeated numerical solutions of a large linear system.

## Acknowledgment

Authors are supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and MITACS. S. Bauer is supported by the German Ministry of Research and Technology BMBF, Grant No. 13N8125 . G. Plank is supported by the Austrian Science Fund FWF, Grant No. R21-N04.

## References

[1] E. Vigmond, F. Aguel, and N. Trayanova, "Computational techniques for solving the bidomain equations in three dimensions." *IEEE Trans Biomed Eng*, vol. 49, no. 11, pp. 1260–9, 2002.

[2] B. Rodriguez and N. Trayanova, "Upper limit of vulnerability in a defibrillation model of the rabbit ventricles." *J Electrocardiol*, vol. 36 Suppl, pp. 51–6, 2003.

[3] H. Shao, K. Sampson, J. Pormann, D. Rose, and C. Henriquez, "A resistor interpretation of general anisotropic cardiac tissue." *Math Biosci*, vol. 187, no. 2, pp. 155–74, 2004.

[4] H. Saleheen and K. Ng, "A new three-dimensional finite-difference bidomain formulation for inhomogeneous anisotropic cardiac tissues." *IEEE Trans Biomed Eng*, vol. 45, no. 1, pp. 15–25, 1998.

[5] N. Hooke, C. Henriquez, P. Lanzkron, and D. Rose, "Linear algebraic transformations of the bidomain equations: implications for numerical methods." *Math Biosci*, vol. 120, no. 2, pp. 127–45, 1994.

[6] R. Weber dos Santos, G. Plank, S. Bauer, and E. Vigmond, "Parallel multigrid preconditioner for the cardiac bidomain model." *IEEE Trans Biomed Eng*, vol. 51, no. 11, pp. 1960–8, 2004.

[7] J. Drouhard and F. Roberge, "Revised formulation of the hodgkin-huxley representation of the sodium current in cardiac cells," *Comp and Biomed Res*, vol. 20, pp. 333–350, 1987.

[8] R. Achar and M. Nakhla, "Simulation of high speed interconnects," *Procedings of IEEE*, vol. 89, pp. 693–728, 2001.

[9] P. Heres and W. Schilders, "Practical issues of model order reduction with Krylov-subspace methods," *Procedings of PIERS, Pisa*, 2004.

[10] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed., 2000.

[11] J. Shewchuk, *An introduction to Conjugate Gradient Method without the agonizing pain*, 1st ed., 1994.