ALGEBRAIC MULTIGRID PRECONDITIONERS FOR THE BIDOMAIN REACTION–DIFFUSION SYSTEM

MICOL PENNACCHIO* AND VALERIA SIMONCINI†

Abstract. The so called *Bidomain system* is possibly the most complete model for the cardiac bioelectric activity. It consists of a reaction–diffusion system, modeling the intra, extracellular and transmembrane potentials, coupled through a nonlinear reaction term with a stiff system of ordinary differential equations describing the ionic currents through the cellular membrane. In this paper we address the problem of efficiently solving the large linear system arising in the finite element discretization of the bidomain model, when a semi-implicit method in time is employed. We analyze the use of structured algebraic multigrid preconditioners on two major formulations of the model, and report on our numerical experience under different discretization parameters and various discontinuity properties of the conductivity tensors. Our numerical results show that the less exercised formulation provides the best overall performance on a typical simulation of the myocardium excitation process.

Key words. reaction–diffusion system, iterative methods, algebraic multigrid, preconditioning

AMS subject classifications. 65F10, 65F15, 65N55, 35K57, 35K65

1. Introduction. The so called *Bidomain system* is possibly the most complete model for the cardiac bioelectric activity, see e.g. [10, 19, 34]. It consists of a reaction–diffusion (R–D) system, modeling the intra, extracellular and transmembrane potentials, coupled through a nonlinear reaction term with a stiff system of ordinary differential equations describing the ionic currents through the cellular membrane. A major bottleneck of the whole procedure is that the excitation process in the myocardium is characterized by different time and space scales, e.g. the small thickness (1–2 mm) of the activation layer versus the much larger size of the cardiac tissue. As a consequence, the numerical solution of the Bidomain system represents a very intensive computational task: realistic three dimensional simulations typically yield discrete problems with at least $O(10^7)$ unknowns, and time steps of the order of 10^{-2} milliseconds or less. To reduce the computational cost, expecially during the excitation process, adaptive techniques and domain decomposition methods have been developed [12, 33, 8, 18, 38, 21]. Adaptivity in space and time may represent a valid solution to reduce the computational cost of the Bidomain system, see e.g. [6]. These techniques have been proven to be successful for problems of moderate sizes and are currently under investigation. Earlier Bidomain studies based on finite differences discretizations can be found in [28, 26, 4].

Most numerical studies now employ semi-implicit methods in time, that only require the solution of linear systems at each time step - as opposed to fully implicit approaches which require nonlinear solvers [16] - and allow larger time steps than explicit schemes. Here we also employ a semi-implicit method, and we face the problem of efficiently solving the very large algebraic linear system arising at each time step. The associated coefficient matrix is inherently singular and block structured, where the blocks are associated with the differential operators appearing in the model. The design of computationally effective iterative solvers for such linear systems calls for the construction of efficient preconditioners, see [36] for a detailed overview of the

*Istituto di Matematica Applicata e Tecnologie Informatiche del C.N.R., via Ferrata, 1 - 27100 Pavia, Italy (micol@imati.cnr.it)

†Dipartimento di Matematica, Università di Bologna, Piazza di Porta S. Donato, 5, I-40127 Bologna, Italy; CIRSA, Ravenna and IMATI-CNR, Pavia, Italy (valeria@dm.unibo.it).

methods. Attempts in the recent literature have employed diagonal preconditioners [29], Symmetric Successive Over Relaxation [20], Block Jacobi preconditioners with incomplete LU factorization (ILU) for each block [8, 35], multigrid [37, 38, 1, 22].

In this work we focus on algebraic multigrid based preconditioners and in particular we study two classes of structured preconditioners that take into account the block form of the coefficient matrix. General Algebraic Multigrid preconditioning has been already applied to the Bidomain system in one of the two formulations we shall adopt, and its effectiveness when compared to other classical methods has been reported; see, e.g., [37, 38, 1, 22].

We recall that the Bidomain system may be written in various forms involving different combinations of its main variables: the intra- and extracellular potentials u_i, u_e and the transmembrane potential $v = u_i - u_e$; see, e.g., [11, 20]. Here we consider two different pairings, the (u_e, v) and the (u_i, u_e) formulations, where the independent variables are u_e, v and u_i, u_e respectively. In spite of the close relationship among the resulting formulations, the associated linear systems provide different challenges, so that the same general preconditioning strategy may lead to surprisingly different performance on the two formulations. We wish to stress that while the (u_e, v) formulation is well accounted for in the current literature, the (u_i, u_e) formulation has been somehow overlooked. This is mostly due to the fact that v is the quantity of chief interest as well as being directly observable from the single cell to the tissue level through optical mapping [5]. Moreover, formulation (u_e, v) has also been preferred for computational reasons: the parabolic and elliptic equations of the formulation can be solved one after the other by means of a block Gauss–Seidel method, see [36, 20, 1]. However, such a "natural" solution process may have misguided the algebraic treatment of the problem. Indeed, our results seem to indicate that formulation (u_i, u_e) is the one that provides a significantly more efficient numerical framework.

The aim of this paper is twofold. On the one hand, we wish to explore the effectiveness of algebraic multigrid based preconditioners, which specifically exploit the structure of the linear system. To the best of our knowledge, a study devoted to the use of block structured multilevel preconditioners on both formulations appears to be new on this problem. On the other hand, we compare the theoretical properties and the performance of similar preconditioning strategies when applied to the two aforementioned formulations. While our study with the (u_e, v) formulation seems to encourage the use of structured algebraic multigrid preconditioners, we shall see that for the (u_i, u_e) formulation this is not necessarily the case. In fact, a standard algebraic multigrid preconditioner on the whole (u_i, u_e) linear system provides the overall best performance, with significantly lower timings than for all other discussed alternatives, on a typical simulation of the myocardium excitation process. These results are in complete agreement with the performance observed in [20], when used with classical preconditioning strategies.

The outline of the paper is as follows. In section 2 we introduce the Bidomain system and two of its mainly used formulations. Space and time discretizations are presented in the same section, leading to the description of the algebraic linear systems to be solved at each time step. By using a unified form for the linear system stemming from the discretization of both formulations, in section 3 we introduce the class of structured preconditioners that we wish to analyze. Section 4 is devoted to the convergence analysis of two classes of structured preconditioners. Finally, Section 5 reports on numerical experiments showing the effectiveness of the algebraic multigrid approach.

Throughout the paper we shall often deal with singular matrices. With a little abuse of notation, we shall denote by A^{-1} the pseudoinverse of the symmetric and singular matrix A. Nonetheless, the singularity of the matrix is taken into account when handling the pseudoinverse.

2. The Bidomain Model. The excitation process in the myocardium is a complex phenomenon characterized by rapid ionic fluxes through the cellular membrane separating the intracellular and the interstitial fluid in the myocardium. Accurate simulations of a complete heartbeat, from the excitation to the recovery phase, have to incorporate realistic fiber geometry, anisotropy of cardiac conductivity and detailed membrane properties. The bidomain model can account for these features (see [10, 19, 34]), and it consists of a Reaction-Diffusion (R-D) system of equations for the intra- and extracellular potential u_i and u_e, coupled through the transmembrane potential $v := u_i - u_e$. In this model the cardiac muscle is viewed as two superimposed anisotropic continuous media, intra (i) and extracellular (e), occupying the same volume and separated from each other by the cell membrane.

The nonlinearity arises through the current-voltage relationship across the membrane which is described by a set of nonlinear ODEs (see [13] for more details). The anisotropic properties of the media are modeled by the intra- and extracellular conductivity tensors $M_i = M_i(\boldsymbol{x})$ and $M_e = M_e(\boldsymbol{x})$ defined as:

$$M_s(\boldsymbol{x}) = \sigma_t^s I + (\sigma_l^s - \sigma_t^s)\ \mathbf{a}(\boldsymbol{x})\mathbf{a}(\boldsymbol{x})^T \qquad s = i, e \tag{2.1}$$

where $\mathbf{a} = \mathbf{a}(\boldsymbol{x})$ is the unit vector tangent to the cardiac fiber at a point $\boldsymbol{x} \in \Omega$, I is the identity matrix and σ_l^s, σ_t^s for $s = i, e$ are the conductivity coefficients along and across fiber, in the (i) and (e) media, assumed constant with $\sigma_l^s > \sigma_t^s > 0$. It can be easily verified that $M_{i,e}$ satisfy the following uniform ellipticity condition:

$$\exists \lambda_m^s, \lambda_M^s > 0, \qquad \lambda_m^s |\boldsymbol{\xi}|^2 \le \boldsymbol{\xi}^T M_s(\boldsymbol{x}) \boldsymbol{\xi} \le \lambda_M^s |\boldsymbol{\xi}|^2 \qquad \forall \boldsymbol{\xi} \in \mathbb{R}^2, \boldsymbol{x} \in \Omega \quad s = i, e \tag{2.2}$$

with $\Omega \in \mathbb{R}^2$ modeling the cardiac tissue.

The R-D system governing the cardiac electric activity may be written in various forms involving different combinations of the variables u_i, u_e, v; see, e.g., [11, 20]. As already mentioned, here we consider two different formulations, the (u_e, v) and the (u_i, u_e) formulations. For both formulations we will deal with a FEM discretization in space and a semi–implicit scheme in time: implicit for the diffusion and explicit for the reaction term.

2.1. (u_i, u_e) **formulation.** The evolution of the intra and extracellular potentials $u_i(\boldsymbol{x}, t)$ and $u_e(x_{\underline{}}t)$ and transmembrane potential $v(\boldsymbol{x}, t) = u_i(\boldsymbol{x}, t) - u_e(\boldsymbol{x}, t)$ is described by the following reaction–diffusion system, characterized by two nonlinear parabolic equations coupled through the reaction term I_{ion} with a system of ODE's describing the evolution of the gating variables and ion concentration:

Given $I_{app} : \Omega\times]0, T[\to \mathbb{R}$ *and* $u_e^0, u_i^0 : \Omega \to \mathbb{R}$, *find* $u_i, u_e : \Omega\times]0, T[\to \mathbb{R}$ *and* $v = u_i - u_e$ *such that*:

$$\begin{cases} c_m \partial_t v - \text{div } M_i \nabla u_i + I_{ion} = I_{app} & \text{in } \Omega\times]0, T[\\ c_m \partial_t v + \text{div } M_e \nabla u_e + I_{ion} = I_{app} & \text{in } \Omega\times]0, T[\\ \mathbf{n}^T M_{i,e} \nabla u_{i,e} = 0 & \text{on } \Gamma\times]0, T[\\ v(\boldsymbol{x}, 0) = 0 & \text{in } \Omega \end{cases} \tag{2.3}$$

where $\Omega \subset \mathbb{R}^2$ models the heart tissue, $\Gamma = \partial\Omega$, $\mathbf{n}$ denotes the outward unit normal to the boundary Γ and I_{app} is an applied current used to initiate the process.

Since the aim of this work is to test the effectiveness of algebraic multigrid preconditioners we mostly focus on the solution of the partial differential equations of the bidomain system. Thus, for simplicity but without losing generality, here we deal with the FitzHugh-Nagumo model for the membrane kinetic. Then I_{ion} is a cubic like function of v: $I_{ion}(v) = \frac{\chi}{c_m} I(v)$ with $I(v) = Gv(1 - v/v_{th})(1 - v/v_p)$, χ the ratio of the membrane area per unit tissue, c_m the surface capacitance of the membrane, G the maximum membrane conductance per unit area and $v_{th},\ v_p$ the threshold and plateau values of v.

For applied current I_{app} satisfying the compatibility condition $\int_\Omega I_{app}\, dx = 0$, system (2.3) uniquely determines v, while the potentials u_i and u_e are defined only up to the same additive time-dependent constant related to the reference potential, see [9].

Space and Time discretization. Let $\mathcal{T}^h$ be a uniform triangulation of Ω and V^h the associated space of P^1 finite elements. We obtain a semidiscrete problem by applying a standard Galerkin procedure. Choosing a finite element basis $\{\varphi_i\}$ for V^h, we denote by

$$C = \left\{ c_{rk} = \int_\Omega \varphi_r \varphi_k \, dx \right\}, \quad A_s = \left\{ a_{r,k} = \sum_{K \in \mathcal{T}_h} \int_K (\nabla \phi_r)^T M_s \nabla \phi_k \ dx \right\} \quad s = i, e$$

the symmetric mass matrix and stiffness matrices and by I^h_{ion} and I^h_{app} the finite element interpolants of I_{ion} and I_{app}, respectively. By using (2.2) and standard finite element arguments, we can assert that A_s, $s = i, e$ are symmetric and positive semidefinite with $A_s\mathbf{e} = 0$; in particular, $a_{k,k}$ for $k = 1, n$ are positive and $a_{k,k} = -\sum_{j \neq k} a_{j,k}$.

The time discretization is performed by a semi–implicit scheme using for the diffusion term the implicit Euler method, while the nonlinear reaction term I_{ion} is treated explicitly. Then the following general algebraic system can be obtained:

$$\mathcal{A}\boldsymbol{\xi}^{k+1} = \mathbf{b} \qquad \text{with} \qquad \mathcal{A} = \begin{bmatrix} C_t + A_i & -C_t \\ -C_t & C_t + A_e \end{bmatrix}, \tag{2.4}$$

$\mathbf{b} = \left[C_t\ \mathbf{v}^k - I^h_{ion}(\mathbf{v}^k) + I^h_{app}; -C_t\ \mathbf{v}^k + I^h_{ion}(\mathbf{v}^k) - I^h_{app} \right]$, $C_t = \frac{c_m}{\tau}\ C$ diagonal with positive diagonal entries, τ the time step, $\mathbf{v}^k = \mathbf{u_i}^k - \mathbf{u_e}^k$ and $\boldsymbol{\xi}^{k+1} = \left[\mathbf{u_i}^{k+1}; \mathbf{u_e}^{k+1} \right]$.

Matrix $\mathcal{A}$ is positive semidefinite, indeed, for $\mathbf{0} \neq \boldsymbol{\xi} = [\mathbf{u_i}; \mathbf{u_e}] \in \mathbb{R}^{2n}$, we have

$$\begin{aligned} \boldsymbol{\xi}^T \mathcal{A} \boldsymbol{\xi} &= \mathbf{u_i}^T C_t \mathbf{u_i} - 2\mathbf{u_e}^T C_t \mathbf{u_i} + \mathbf{u_e}^T C_t \mathbf{u_e} + \mathbf{u_i}^T A_i \mathbf{u_i} + \mathbf{u_e} A_e \mathbf{u_e} \\ &= (\mathbf{u_i} - \mathbf{u_e})^T C_t (\mathbf{u_i} - \mathbf{u_e}) + \mathbf{u_i}^T A_i \mathbf{u_i} + \mathbf{u_e}^T A_e \mathbf{u_e} \geq 0, \end{aligned}$$

since A_s, $s = i, e$ are positive semidefinite and C_t is positive definite. Moreover, for $\mathbf{e}$ the vector of all ones, $\mathcal{A}[\mathbf{e}; \mathbf{e}] = \mathbf{0}$ and the system is consistent, in that $\mathbf{b}$ has zero mean, that is $\mathbf{e}^T\mathbf{b} = \mathbf{0}$. As in the continuous model, $\mathbf{v}^k$ is uniquely determined, while $\mathbf{u_i}^k$ and $\mathbf{u_e}^k$ are determined only up to the same additive time-dependent constant.

2.2. (u_e, v) **formulation.** In system (2.3) the bidomain model was formulated in terms of the potential fields u_i and u_e but it can be equivalently expressed in terms of the transmembrane and extracellular potentials v and u_e. Indeed, adding the two evolution equations of the system (2.3) and substituting $v = u_i - u_e$, we obtain an elliptic equation in the unknown (u_e, v), i.e. the following equivalent formulation of the anisotropic bidomain model:

find $(v(\boldsymbol{x},t), u_e(\boldsymbol{x},t))$, $\boldsymbol{x} \in \Omega$, $t \in [0,T]$ *such that*

$$\begin{cases} c_m \partial_t v - \text{div } M_i \nabla v + I_{ion} = \text{div } M_i \nabla u_e + I_{app} & \text{in } \Omega \times]0,T[\\ -\text{div } M \nabla u_e = \text{div } M_i \nabla v & \text{in } \Omega \times]0,T[\\ \mathbf{n}^T M_i \nabla v = 0, \qquad \mathbf{n}^T M \nabla u_e = 0 & \text{on } \Gamma \times]0,T[\\ v(\boldsymbol{x},0) = 0 & \text{in } \Omega. \end{cases} \tag{2.5}$$

with $M = M_i + M_e$ bulk conductivity tensor.

The above system differs from problem (2.3) in that the change of variable allows us to replace the second parabolic equation with an elliptic equation, thus loosing the degenerate temporal structure of system (2.3). This approach was usually preferred because the two equations can be solved one after the other.

Proceeding as before, i.e. using a finite element discretization in space and a semi–implicit scheme in time, we get:

$$\mathcal{M}\boldsymbol{\xi}^{k+1} = \mathbf{b} \qquad \text{with} \qquad \mathcal{M} = \begin{bmatrix} C_t + A_i & A_i \\ A_i & (A_i + A_e) \end{bmatrix}, \tag{2.6}$$

with $\mathbf{b} = \left[C_t\ \mathbf{v}^k - I_{ion}^h(\mathbf{v}^k) + I_{app}^h; \mathbf{0}\right]$, $\mathbf{v}^k = \mathbf{u_i}^k - \mathbf{u_e}^k$, $\boldsymbol{\xi}^{k+1} = \left[\mathbf{v}^{k+1}; \mathbf{u_e}^{k+1}\right]$. As before, we can prove that the symmetric matrix $\mathcal{M}$ is positive semidefinite: let $\mathbf{0} \neq \boldsymbol{\xi} = [\mathbf{v}; \mathbf{u_e}] \in \mathbb{R}^{2n}$, then

$$\begin{aligned} \boldsymbol{\xi}^T \mathcal{M} \boldsymbol{\xi} &= \mathbf{v}^T C_t \mathbf{v} + \mathbf{v}^T A_i \mathbf{v} + \mathbf{u}^T (A_i + A_e)\mathbf{u} + 2\mathbf{v}^T A_i \mathbf{u} \\ &= \mathbf{v}^T C_t \mathbf{v} + (\mathbf{v}+\mathbf{u})^T A_i (\mathbf{v}+\mathbf{u}) + \mathbf{u}^T A_e \mathbf{u} \geq 0 \end{aligned}$$

since A_s, $s = i, e$ are positive semidefinite and C_t is positive definite. Moreover, $\mathcal{M}[\mathbf{0}; \mathbf{e}] = 0$.

At each time step we thus have to solve the large linear system (2.6) or (2.4), whose conditioning considerably worsens as the problem dimension increases, resulting in an unacceptable increase in the computational costs of the whole simulation. In this context, preconditioning is therefore mandatory. It is also important to realize that the possibly high cost of constructing an effective preconditioner may be acceptable when compared to the total CPU time required by the entire simulation, if the preconditioner is generated once for all.

Classically, the system with $\mathcal{M}$ was solved by means of a nested iteration that can be explicitly stated as a block Gauss–Seidel method involving the two diagonal blocks.

3. Block Preconditioners. Both formulations are characterized by a 2×2 block form, which for brevity we denote by

$$\mathcal{B} = \begin{pmatrix} B_{11} & B_{12} \\ B_{12} & B_{22} \end{pmatrix}, \tag{3.1}$$

with B_{11}, B_{22} positive (semi)definite. Note that all matrices are square and symmetric. It is therefore natural to derive preconditioners that exploit this structure, as is the case in various multilevel methods (see, e.g., [30], [32]) as well as in saddle point problems [3].

Here we analyze two symmetric structured preconditioners, namely block diagonal and block factorized preconditioners, which have been widely used in the literature in a number of contexts. To this end, we consider

$$\mathcal{P}_d = \text{blockdiag}(K, D), \tag{3.2}$$

where K is a symmetric and positive definite approximation to the (1,1) block B_{11}, while D is a symmetric and positive definite approximation either to the Schur complement $B_{22} - B_{12}B_{11}^{-1}B_{12}$, or to the (2,2) block B_{22}. The application of this preconditioner at each iteration entails one solve with each of the two matrices K and D.

We also consider the more expensive, but possibly more effective block factorized preconditioner

$$\mathcal{P}_f = \begin{bmatrix} I & O \\ B_{12}K^{-1} & I \end{bmatrix} \begin{bmatrix} K & B_{12} \\ O & D \end{bmatrix},$$

where K and D have the same meaning as before. The application of this factorized preconditioner corresponds to the solution of two block triangular systems, that is

$$\begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{bmatrix} = \mathcal{P}_f^{-1} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} K^{-1} & -K^{-1}B_{12}D^{-1} \\ O & D^{-1} \end{bmatrix} \begin{bmatrix} I & O \\ -B_{12}K^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}.$$

This may be performed with the following steps
$\boldsymbol{z}_1 = K^{-1}\mathbf{v}_1$
$\boldsymbol{y}_2 = D^{-1}(\mathbf{v}_2 - B_{12}\boldsymbol{z}_1)$
$\boldsymbol{y}_1 = \boldsymbol{z}_1 - K^{-1}(B_{12}\boldsymbol{y}_2)$
showing that the application of $\mathcal{P}_f^{-1}$ requires two solves with K and one solve with D, together with two (cheap) multiplications with the sparse matrix B_{12}.

In the following sections we discuss a-priori convergence results on the convergence of CG when $\mathcal{P}_d$ or $\mathcal{P}_f$ are used as preconditioners, and we compare them with other alternatives. The choice of the block matrices K and D are based on their approximation to some corresponding blocks in the original matrix. More precisely, for two symmetric positive semidefinite matrices A and B, we will use the notation $A \leq B$ if $\mathbf{v}^T A\mathbf{v} \leq \mathbf{v}^T B\mathbf{v}$. Therefore, we say that B is a good preconditioner for A if there exist two positive constants c_1, c_2, as close as possible to one, such that $c_1 B \leq A \leq c_2 B$. The two matrices A and B are spectrally equivalent if the constants c_1, c_2 do not depend on specific discretization parameters, such as the grid size.

4. Convergence analysis. The Conjugate Gradient (CG) method can be used to solve the large linear systems $\mathcal{B}\boldsymbol{\xi} = \mathbf{b}$ when $\mathcal{B}$ is symmetric and positive definite. In general, the possible singularity of $\mathcal{B}$ is not harmful for CG, as long as the system is consistent, that is, the right-hand side does not have any component onto the null space of $\mathcal{B}$. After j iterations of CG, the error in the approximation, measured in the energy norm, can be bounded in the following well known manner (see, e.g., [27])

$$\|\boldsymbol{\xi}_j - \boldsymbol{\xi}\|_{\mathcal{B}} \leq \left(\frac{\sqrt{\mathrm{cond}(\mathcal{B})} - 1}{\sqrt{\mathrm{cond}(\mathcal{B})} + 1} \right)^j, \tag{4.1}$$

where $\mathrm{cond}(\mathcal{B})$ is the spectral condition number of the coefficient matrix, that is, $\mathrm{cond}(\mathcal{B}) = \lambda_{\max}(\mathcal{B})/\lambda_{\min}(\mathcal{B})$, in which $\lambda_{\max}(\mathcal{B})$, $\lambda_{\min}(\mathcal{B})$ are the largest and smallest nonzero eigenvalues of $\mathcal{B}$. Note that from this estimate, a bound for the Euclidean norm of the residual can also be obtained, since $\|\mathbf{b} - \mathcal{B}\boldsymbol{\xi}_j\| \leq \lambda_{\max}(\mathcal{B})^{1/2} \|\boldsymbol{\xi}_j - \boldsymbol{\xi}\|_{\mathcal{B}}$. The right-hand side in (4.1) provides a bound on the *worst case* convergence scenario for the given linear system. In particular, it may give a good estimate of the convergence *rate* of the method, rather than an accurate account of the actual convergence history. This fact will be shown in the following sections.

To speed up convergence, preconditioning is employed so as to lower the condition number of the preconditioned matrix. In this section we illustrate some condition number estimates of the preconditioned matrix, when the structured preconditioners of the previous sections are employed.

We first recall some general results that can be appropriately adapted to the analysis of the preconditioners $\mathcal{P}_d$, $\mathcal{P}_f$. To this end, we use once again the generic matrix notation in (3.1). We introduce the following constant (C.B.S. constant)

$$\gamma^2 = \sup_{\mathbf{v}\in\mathbb{R}^n\setminus N(B_{22})} \frac{\mathbf{v}^T B_{12} B_{11}^{-1} B_{12}\mathbf{v}}{\mathbf{v}^T B_{22}\mathbf{v}}, \tag{4.2}$$

where $N(B_{22})$ is the null space of the possibly singular matrix B_{22}. The following result holds for the condition number of the block diagonal preconditioner[1]

PROPOSITION 4.1. *[2, Theorem 9.3] Let γ be defined in (4.2). Assume that the matrices K and D in (3.2) are such that there exist positive constants $\alpha_1, \alpha_2, \beta_1, \beta_2$ satisfying*

$$\alpha_1 B_{11} \le K \le \alpha_2 B_{11}, \quad \beta_1 B_{22} \le D \le \beta_2 B_{22},$$

with $\alpha_2 \ge \beta_2$. Then

$$\mathrm{cond}(\mathcal{P}_d^{-1}\mathcal{B}) \le \frac{\alpha_2}{\alpha_1(1-\gamma^2)}\phi(\alpha_1/\beta_1)\phi(\beta_2/\alpha_2)$$

where $\phi(\tau) = \frac{1}{2}(1+\tau) + \sqrt{\frac{1}{4}(1-\tau)^2 + \tau\gamma^2}$.

Note that $\phi(\tau) \to 1+\gamma$ for $\tau \to 1$, while $\phi(\tau) \to 1$ as $\tau \to 0$. Function ϕ behaves linearly in τ as long as γ is close to unit. For γ smaller, ϕ grows much more slowly.

For $K = B_{11}$ and $D = B_{22}$ it holds that $\mathrm{cond}(\mathcal{P}_d^{-1}\mathcal{B}) = (1+\gamma)/(1-\gamma)$ ([2, Corollary 9.4]). Therefore, if K and D are very good approximations to the corresponding matrices B_{11} and B_{22}, the performance of PCG is still driven by γ, which is problem dependent. In particular, if γ is very close to one, convergence can still be slow, and other structured preconditioners should be considered; see also the discussion in [17].

The following result holds for the condition number of the block factorized preconditioner.

PROPOSITION 4.2. *[2, Theorem 9.5] Assume there exist positive constants α_1, α_2, δ_1, δ_2 such that*

$$\alpha_1 B_{11} \le K \le \alpha_2 B_{11}, \quad \delta_1 B_{22} \le S \le \delta_2 B_{22},$$

where $S = D + B_{12}K^{-1}B_{12}$, and $\alpha_2 \ge 1 \ge \alpha_1 > \gamma^2$, $\delta_2 \ge 1 \ge \delta_1 > \gamma^2$. Let $\phi = \phi(\tau)$ be as defined in Proposition 4.1. Then

$$\lambda_{\min}(\mathcal{P}_f^{-1}\mathcal{B}) \ge \left(1 + \frac{\max\{\alpha_2,\delta_2\}-1}{1-\gamma^2}\phi(r_2)\right)^{-1}$$

where $r_2 = \min\{\frac{\alpha_2-1}{\delta_2-1}, \frac{\delta_2-1}{\alpha_2-1}\}$ and $\alpha_2 > 1$ and/or $\delta_2 > 1$.

Moreover,

$$\lambda_{\max}(\mathcal{P}_f^{-1}\mathcal{B}) \le \left(1 - \frac{1-\min\{\alpha_1,\delta_1\}}{1-\gamma^2}\phi(r_1)\right)^{-1}$$

[1]Here and in the following, the notation $B \le A$ with A singular, silently assumes that the relation is required to hold only in the range of A.

where $r_1 = \min\{\frac{1-\alpha_1}{1-\delta_1}, \frac{1-\delta_1}{1-\alpha_1}\}$ *and* $\alpha_1 < 1$ *and/or* $\delta_1 < 1$.

In particular, if $K = B_{11}$, then $\mathrm{cond}(\mathcal{P}_f^{-1}\mathcal{B}) \leq (\delta_2 - \gamma^2)/(\delta_1 - \gamma^2)$ [2, Case 9.1, p.386]. Other special cases are discussed in the same reference.

4.1. Specialized analysis for the two formulations. The results of the previous section may be specialized to the two matrix formulations, by replacing the matrix $\mathcal{B}$ above with the matrix in (2.6) and in (2.4), for the (u, v) and (u_i, u_e) formulations, respectively.

4.1.1. The (u, v) formulation. In this setting, the CBS constant is given as

$$\gamma^2 = \sup_{\mathbf{v} \in \mathbb{R}^n \setminus N(A_i + A_e)} \frac{\mathbf{v}^T A_i (C_t + A_i)^{-1} A_i \mathbf{v}}{\mathbf{v}^T (A_i + A_e)\mathbf{v}}. \tag{4.3}$$

Next lemma ensures that $\gamma < 1$ in the (u, v) formulation and provides a bound of γ^2 in terms of the eigenvalues of the pair (A_e, A_i). We recall here that $N(A_i + A_e) = N(A_i) = N(A_e)$.

LEMMA 4.3. *With the previous notation,* $\gamma^2 \leq (1 + \lambda_{\min}(A_e, A_i))^{-1} < 1$*, where* $\lambda_{\min}(A_e, A_i)$ *is the smallest nonzero eigenvalue of* $A_e A_i^{-1}$.

Proof. We have $C_t + A_i > A_i$, so that $(C_t + A_i)^{-1} < A_i^{-1}$. Moreover[2], $A_i A_i^{-1} A_i = A_i$. For any $\mathbf{v} \in \mathbb{R}^n \setminus N(A_i + A_e)$, $\mathbf{v} \neq 0$ we have

$$\frac{\mathbf{v}^T A_i (C_t + A_i)^{-1} A_i \mathbf{v}}{\mathbf{v}^T (A_i + A_e)\mathbf{v}} \leq \frac{\mathbf{v}^T A_i \mathbf{v}}{\mathbf{v}^T (A_i + A_e)\mathbf{v}} = \frac{\boldsymbol{w}^T \boldsymbol{w}}{\boldsymbol{w}^T (I + A_i^{-\frac{1}{2}} A_e A_i^{-\frac{1}{2}})\boldsymbol{w}} \leq \frac{1}{1 + \lambda_{\min}(A_e, A_i)},$$

where $\boldsymbol{w} = A^{1/2}\mathbf{v}$ and $A_i^{-1/2} = (A_i^{-1})^{1/2}$. ∎

The result shows that γ^2 is bounded by a quantity that only depends on the conductivity tensors of the two stiffness matrices, and not on the grid. Indeed, it can be shown that for $0 \neq \mathbf{v} \in \mathbb{R}^n \setminus N(A_i)$, the two stiffness matrices are related as $c_1 \mathbf{v}^T A_e \mathbf{v} \leq \mathbf{v}^T A_i \mathbf{v} \leq c_2 \mathbf{v}^T A_e \mathbf{v}$, with $c_1 = \sigma_t^i / \sigma_t^e$ and $c_2 = (\sigma_l^i - \sigma_t^i)/(\sigma_l^e - \sigma_t^e)$, independently of the mesh.

In this formulation, the spectral scalars α_i, β_i are defined by the following inequalities involving the approximation symmetric matrices K and D:

$$\alpha_1(A_i + C_t) \leq K \leq \alpha_2(A_i + C_t), \quad \beta_1(A_i + A_e) \leq D \leq \beta_2(A_i + A_e). \tag{4.4}$$

Table 4.1 reports the values of these parameters for the example in section 5 when an Algebraic Multigrid method (AMG) is used to approximate the blocks $A_i + C_t$ and $A_i + A_e$, The table also reports the values of γ^2. We refer to section 5 for additional information on AMG preconditioning.

The numbers clearly show independence of the meshsize. We notice that the condition $\alpha_2 \geq \beta_2$ in Proposition 4.1 does not hold in this case. However, a closer inspection of the proof of Theorem 9.3 in [2] reveals that for the result to hold, it is sufficient that the following weaker conditions on α_2, β_2 hold (cf. formula (9.15) in [2])

$$\gamma^2 \leq \frac{1}{2}(1 - \frac{\alpha_2}{\beta_2}) + \sqrt{\frac{1}{4}(1 - \frac{\alpha_2}{\beta_2})^2 + \frac{\alpha_2}{\beta_2}\gamma^2} \leq 1.$$

[2]Note that the equality holds also in the singular case, where A_i^{-1} is the pseudoinverse of A_i. [15, p.422]

n	α_1	α_2	β_1	β_2	γ^2
2705	1	1.0164	1	1.4233	0.3856
10657	1	1.0785	1	1.6261	0.4282
42305	1	1.1609	1	1.7084	0.4406
168577	1	1.2054	1	1.8278	0.4435

TABLE 4.1
(u, v) formulation. Values of $\alpha_1, \alpha_2, \beta_1, \beta_2$ for different problem sizes, and values of γ^2 in (4.3).

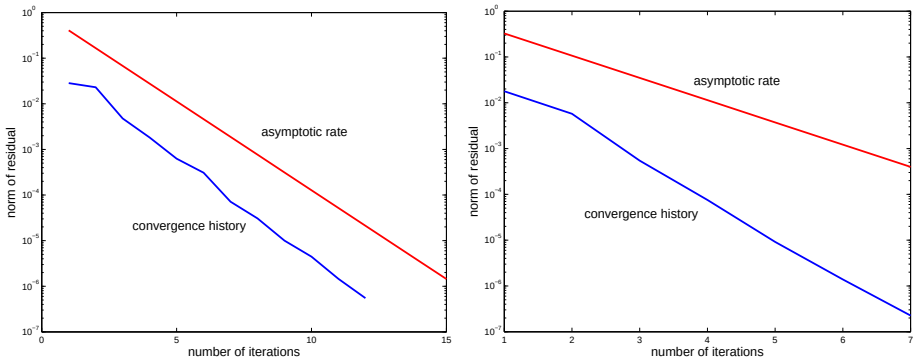

FIG. 4.1. *(u, v) formulation. Convergence history and asymptotic rate of the P_d and P_f preconditioner for n=10657.*

It can be easily verified that these two inequalities hold with the data in Table 4.1.

For the block diagonal preconditioner, Proposition 4.1 with the blocks of this formulation yields

$$\mathrm{cond}(\mathcal{P}_d^{-1}\mathcal{B}) \leq \frac{\alpha_2}{\alpha_1(1-\gamma^2)}\phi(\alpha_1/\beta_1)\phi(\beta_2/\alpha_2). \tag{4.5}$$

The convergence history (residual norm) for $n = 10657$ of CG preconditioned by $\mathcal{P}_d$ is reported in the left plot of Figure 4.1. The plot also shows the approximate asymptotic convergence rate obtained by using the right-hand side in the inequality (4.5), in the classical CG bound (4.1). The agreement is quite satisfactory.

For the factorized preconditioner, we next show that if K and D are specifically chosen, as in (4.4), then the hypothesis on S of Proposition 4.2 is satisfied.

LEMMA 4.4. *Let $S = D + A_i K^{-1} A_i$. If*

$$\alpha_1(C_t + A_i) \leq K \leq \alpha_2(C_t + A_i), \quad \beta_1(A_i + A_e) \leq D \leq \beta_2(A_i + A_e),$$

then $\delta_1(A_i + A_e) \leq S \leq \delta_2(A_i + A_e)$, with $\delta_1 = \beta_1$ and $\delta_2 = \beta_2 + \alpha_1^{-1}$.

Proof. For $\mathbf{v} \neq 0$, the lower bound is readily obtained as $\mathbf{v}^T S\mathbf{v} = \mathbf{v}^T D\mathbf{v} + \mathbf{v}^T A_i K^{-1} A_i \mathbf{v} \geq \beta_1 \mathbf{v}^T(A_e + A_i)\mathbf{v}$, with $\delta_1 = \beta_1$. To get the upper bound, for $\mathbf{v} \neq 0$ and recalling from Lemma 4.3 that $A_i(A_i + C_t)^{-1}A_i \leq A_i$ we write

$$\begin{aligned}\mathbf{v}^T S\mathbf{v} = \mathbf{v}^T D\mathbf{v} + \mathbf{v}^T A_i K^{-1} A_i \mathbf{v} &\leq \beta_2 \mathbf{v}^T(A_e + A_i)\mathbf{v} + \frac{1}{\alpha_1}\mathbf{v}^T A_i(A_i + C_t)^{-1}A_i\mathbf{v} \\ &\leq \beta_2 \mathbf{v}^T(A_e + A_i)\mathbf{v} + \frac{1}{\alpha_1}\mathbf{v}^T A_i\mathbf{v} \leq \delta_2 \mathbf{v}^T(A_e + A_i)\mathbf{v},\end{aligned}$$

where $\delta_2 = \beta_2 + \alpha_1^{-1}$. □

Lemma 4.4 shows that in this formulation, it is sufficient to obtain a good approximation D to the (2,2) block, and S will also be a good approximation to the matrix used in Proposition 4.2.

With the hypotheses of the lemma above, and under additional hypotheses on the parameters, Proposition 4.2 ensures that the following bound on the condition number of the preconditioned matrix holds:

$$\mathrm{cond}(\mathcal{P}_f^{-1}\mathcal{B}) \leq \frac{1 + \frac{\max\{\alpha_2,\delta_2\}-1}{1-\gamma^2}\phi(r_2)}{1 - \frac{1-\min\{\alpha_1,\delta_1\}}{1-\gamma^2}\phi(r_1)}. \tag{4.6}$$

Clearly, if α_i, δ_i are mesh independent, so is $\mathrm{cond}(\mathcal{P}_f^{-1}\mathcal{B})$.

The right plot of Figure 4.1 shows the actual residual norm curve of CG preconditioned with $\mathcal{P}_f$ for the problem of size $n = 10657$. The asymptotic rate was obtained by using (4.6) in (4.1), with the parameters in Table 4.1. The agreement is once again quite satisfactory.

4.1.2. The (u_i, u_e) formulation. In this case the CBS constant is given as

$$\gamma^2 = \sup_{\mathbf{v}\in\mathbb{R}^n\setminus N(A_e)} \frac{\mathbf{v}^T C_t (C_t + A_i)^{-1} C_t \mathbf{v}}{\mathbf{v}^T (C_t + A_e)\mathbf{v}}. \tag{4.7}$$

We notice that the condition $\mathbf{v} \in \mathbb{R}^n \setminus N(A_e)$ would not be requested to ensure a nonzero denominator. However, due to the consistency of the linear system, vectors in $N(A_e)$ are not involved in the computation, so that the constraint $\mathbf{v} \in \mathbb{R}^n \setminus N(A_e)$ is verified in practice, assuming exact arithmetic. It is interesting to observe that without the condition $\mathbf{v} \in \mathbb{R}^n \setminus N(A_e)$, we would have obtained $\gamma = 1$, readily derived for $\mathbf{v} \in N(A_e) = N(A_i)$.

Next lemma ensures that $\gamma < 1$ in the (u_i, u_e) formulation.

LEMMA 4.5. *With the previous notation, $\gamma^2 \leq (1 + \lambda_{\min}(A_e, C_t))^{-1} < 1$, where $\lambda_{\min}(A_e, C_t)$ is the smallest nonzero eigenvalue of $A_e C_t^{-1}$.*

Proof. We have $C_t + A_i \geq C_t$, so that $(C_t + A_i)^{-1} \leq C_t^{-1}$. For any $\mathbf{v} \in \mathbb{R}^n \setminus N(A_e)$, $\mathbf{v} \neq 0$ we have

$$\frac{\mathbf{v}^T C_t (C_t + A_i)^{-1} C_t \mathbf{v}}{\mathbf{v}^T (C_t + A_e)\mathbf{v}} \leq \frac{\mathbf{v}^T C_t \mathbf{v}}{\mathbf{v}^T (C_t + A_e)\mathbf{v}} = \frac{\boldsymbol{w}^T \boldsymbol{w}}{\boldsymbol{w}^T (I + C_t^{-\frac{1}{2}} A_e C_t^{-\frac{1}{2}})\boldsymbol{w}} \leq \frac{1}{1 + \lambda_{\min}(A_e, C_t)},$$

where $\boldsymbol{w} = C_t^{1/2}\mathbf{v}$. □

We notice that the bound is quite accurate. For instance, for the smallest grid in section 5 (A_i of size 2705), we found $\gamma = 0.99913$ while $(1 + \lambda_{\min}(A_e, C_t))^{-1} = 0.99978$. Note also that in this example, γ is very close to unit (cf. also Table 4.2), predicting slow convergence of the preconditioned method even in the case of optimal preconditioning, namely for $\alpha_i = \beta_i = 1$, $i = 1, 2$. Since C_t is the mass matrix, it is also quite clear that the bound for γ may depend on the mesh.

The following result shows that the hypothesis on S in Proposition 4.2 is satisfied in the (u_i, u_e) formulation by appropriately choosing D.

LEMMA 4.6. *Let $S = D + C_t K^{-1} C_t$. If*

$$\alpha_1(C_t + A_i) \leq K \leq \alpha_2(C_t + A_i), \quad \beta_1(C_t + A_e) \leq D \leq \beta_2(C_t + A_e),$$

then $\delta_1(C_t + A_e) \leq S \leq \delta_2(C_t + A_e)$, with $\delta_1 = \beta_1$ and $\delta_2 = \beta_2 + \alpha_1^{-1}$.

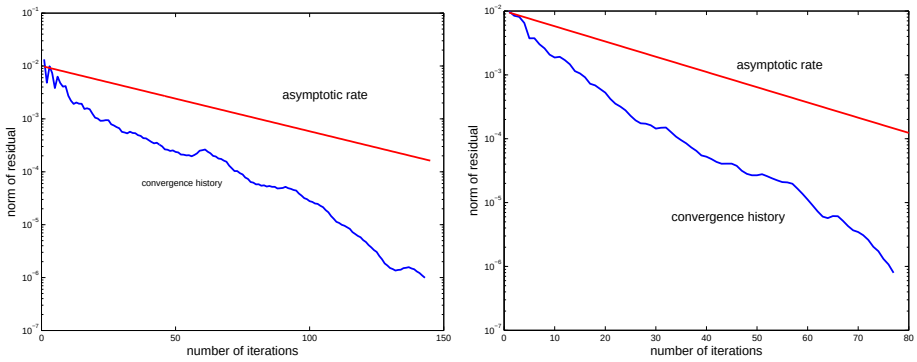

FIG. 4.2. *(u_i, u_e) formulation. Convergence history and asymptotic rate of the P_d and P_f preconditioner for n=10657.*

Proof. The proof is similar to that of Lemma 4.4. For $\mathbf{v} \neq 0$, the lower bound is readily obtained as $\mathbf{v}^T S\mathbf{v} = \mathbf{v}^T D\mathbf{v} + \mathbf{v}^T C_t K^{-1} C_t \mathbf{v} \geq \beta_1 \mathbf{v}^T (A_e + C_t)\mathbf{v}$, with $\delta_1 = \beta_1$. To get the upper bound, for $\mathbf{v} \neq 0$ and recalling that $(A_i + C_t)^{-1} \leq C_t^{-1}$ (cf. proof of Lemma 4.5), we write

$$\begin{aligned}\mathbf{v}^T S\mathbf{v} &= \mathbf{v}^T D\mathbf{v} + \mathbf{v}^T C_t K^{-1} C_t \mathbf{v} \leq \beta_2 \mathbf{v}^T (A_e + C_t)\mathbf{v} + \frac{1}{\alpha_1} \mathbf{v}^T C_t (A_i + C_t)^{-1} C_t \mathbf{v} \\ &\leq \beta_2 \mathbf{v}^T (A_e + C_t)\mathbf{v} + \frac{1}{\alpha_1} \mathbf{v}^T C_t \mathbf{v} \leq \delta_2 \mathbf{v}^T (A_e + C_t)\mathbf{v},\end{aligned}$$

where $\delta_2 = \beta_2 + \alpha_1^{-1}$. ∎

In particular, if S is exact, that is $S = C_t + A_e$, then $\delta_1 = \delta_2 = 1$ and Proposition 4.2 ensures that

$$\mathrm{cond}(\mathcal{P}_f^{-1}\mathcal{A}) \leq \frac{\alpha_2 - \gamma^2}{\alpha_1 - \gamma^2}. \tag{4.8}$$

In Table 4.2 we report the relevant parameters of this formulation when K and D are computed with the AMG preconditioner available in the PIFISS software (see section 5 for more details).

n	α_1	α_2	β_1	β_2	γ^2
2705	1	1.0164	1	1.0164	0.99912931
10657	1	1.0785	1	1.0734	0.99912954
42305	1	1.1609	1	1.1036	0.99912961
168577	1	1.2054	1	1.2509	0.99912962

TABLE 4.2

(u_i, u_e) formulation. Values of $\alpha_1, \alpha_2, \beta_1, \beta_2$ for different problem sizes, and values of γ^2 in (4.7).

REMARK 4.7. The values of γ appear to be very close to one already for quite coarse meshes (cf. Table 4.2). As a result, even assuming that it is affordable to explicitly approximate the Schur complement matrix $D \approx S - C_t(C + A_i)^{-1} C_t$, convergence may be slow. These considerations are confirmed by our numerical experiments. For this reason, in the case of this formulation we have explored various alternatives for

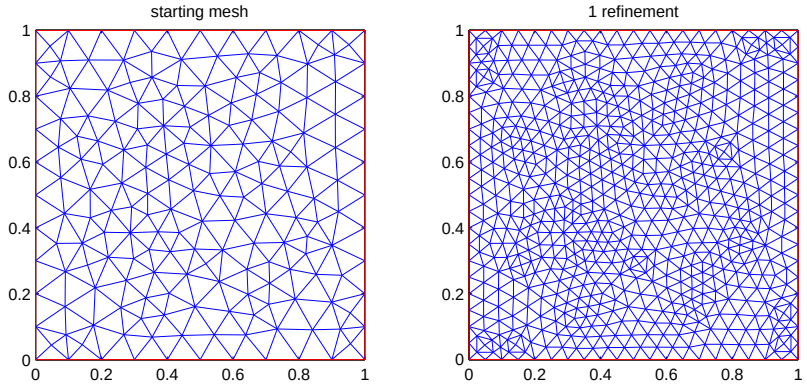

FIG. 5.1. *Delaunay mesh: starting mesh (left) - 1 refinement (right)*

D. Our experiments of section 5 reveal that nonoptimal choices of D can still provide us with fast convergence, although a mild dependence on the mesh remains. This is revealed by the low but increasing number of iterations.

Figure 4.2 shows the convergence history (Euclidean norm of the residual) of CG on the problem for $n = 10657$ when the preconditioners $\mathcal{P}_d$ (left) and $\mathcal{P}_f$ (right) are used. In both plots, the asymptotic convergence estimated with the computed parameters is also reported. In both cases, the asymptotics are able to capture the worst convergence phase of the actual convergence history.

5. Numerical results. The computational experiments were carried out to compare the different formulations and solvers introduced in the previous sections. We deal with a square domain $\Omega = [0,1] \times [0,1]$ representing a block of myocardium with cardiac fibers parallel to a diagonal of the square and the following conductivity coefficients: $\sigma_l^e = 2.5 \times 10^{-3}$, $\sigma_t^e = 1.25 \times 10^{-3}$, $\sigma_l^i = 2. \times 10^{-3}$ $\sigma_t^i = 4.16 \times 10^{-4}$ (Ω^{-1} cm^{-1}).

We build our meshes on Ω by using a Delaunay triangulation algorithm. We fix a starting mesh and we build the subsequent meshes by using a regular refinement where all of the specified triangles are divided into four triangles of the same shape. See Figure 5.1 for a typical mesh refinement. The number of nodes of the meshes considered is $2n$ with $n \in \{2705, 10657, 42305, 168577, 673025\}$ and the corresponding mesh size $h \in \{3.13 \cdot 10^{-2}, 1.57 \cdot 10^{-2}, 7.8 \cdot 10^{-3}, 3.9 \cdot 10^{-3}, 1.9 \cdot 10^{-3}\}$ whereas the time step τ was chosen to be equal to $4 \cdot 10^{-2}$ msec. With these time and space steps we can obtain stable and accurate results as shown by the validation carried out in [7].

All experiments correspond to a typical temporal instant in the time step evolution, so that the right–hand side includes information generated during the previous time steps. All computations were performed with Matlab 7.4.0 (R2007a) [14] on a Dell Power Edge 2800, 16 Gb RAM, Intel Xeon dual core 64 bit 3.4 GHz and 2Mb L2 cache.

We are interested in comparing the performance of the block structured preconditioners described in the preceding sections, when Algebraic MultiGrid (AMG) is used to build the matrices K and D to approximate the corresponding blocks. Before approximating each single blocks with various preconditioners, we reorder each block matrices of $\mathcal{B}$ by using the matlab function *symrcm*. We employ the AMG

n	P_d	P_f	$\mathcal{P}_{whole}$	$\mathcal{P}_d^{(ic)}$
2705	0.36 (10)	0.59 (5)	0.88 (12)	0.15 (8)
10657	1.95 (12)	1.99 (7)	9.36 (29)	0.58 (14)
42305	10.37 (15)	13.73 (7)	60.33 (55)	4.15 (27)
168577	51.67 (16)	42.84 (10)	491.42 (109)	45.82 (70)
673025	232.16 (16)	167.10 (10)	6025.68 (161)	549.24 (204)

TABLE 5.1

(u_e, v) formulation: CPU time and number of iterations (in parenthesis) for P_d, P_f, $\mathcal{P}_{whole}$ and $\mathcal{P}_d^{(ic)}$.

code available within the Matlab package PIFISS ([25, 23, 24]), which is the original AMG1R5 algorithm, as once distributed by the SCAI Institute. The good performance of the code on 2D stiffness matrices with constant coefficients has been largely verified; see, e.g., [31]. For this reason, the algorithm was used as a black box; Gauss-Seidel smoothing was used in all instances. In most cases, the multilevel method is built on originally singular matrices. To increase the robustness of the preconditioning strategy, in some cases we generated the preconditioner by using a shifted (nonsingular) matrix, with a shift equal to $\varepsilon^{1/2}$, where $\varepsilon \approx 10^{-16}$ is Matlab machine precision.

To test the effectiveness of the AMG block preconditioners, we report results on both formulations when Incomplete Cholesky factorizations (IC) are used to generate K and D in the block diagonal preconditioner, and we call the resulting preconditioner $\mathcal{P}_d^{(ic)}$. In this case, the obtained blocks are not spectrally equivalent to the target matrices, therefore the condition number of the preconditioned matrix may depend on the mesh parameters. We also consider preconditioning the whole matrix $\mathcal{B}$ ($\mathcal{B} = \mathcal{A}$ or $\mathcal{B} = \mathcal{M}$, depending on the formulation) with AMG, and we call this preconditioner $\mathcal{P}_{whole}$. In this last case, the entries of the matrix were reordered using Symmetric reverse Cuthill-McKee permutation (Matlab function `symrcm`), which lead to better timings.

To simplify the notation, we denote by $\mathcal{P}_f(K, D)$ the factorized matrix for the specific choices of K and D.

Formulation (u, v): we consider solving the system (2.6) with the following preconditioners, where W in the last row is defined as

$$W = (A_i + A_e) + A_i \mathrm{diag}(C_t + A_i)^{-1} A_i. \tag{5.1}$$

Prec.	K	D	other	comments
$\mathcal{P}_d$	AMG($C_t + A_i$)	AMG($A_i + A_e$)		cf. Prop. 4.1
$\mathcal{P}_f$	AMG($C_t + A_i$)	AMG($A_i + A_e$)		cf. Prop. 4.2
$\mathcal{P}_{whole}$			AMG($\mathcal{M}$)	$\mathcal{M}$ as in (2.6)
$\mathcal{P}_d^{(ic)}$	IC($C_t + A_i$,10^{-2})	IC(W,$3 \cdot 10^{-3}$)		W as in (5.1)

The results, CPU times in seconds and number of CG iterations in parentheses, to obtain a reduction of the residual norm by a factor of 10^{-6}, are reported in Table 5.1.

We readily observe that AMG on the whole matrix, namely $\mathcal{P}_{whole}$, is not competitive, and the number of iterations significantly increases with the problem size. This latter shortcoming is also true for the Incomplete Cholesky preconditioner, as expected. We notice, however, that $\mathcal{P}_d^{(ic)}$ provides the best timings on the smaller

problems. Both preconditioners based on structured AMG preconditioning enjoy mesh independence, with $\mathcal{P}_f$ becoming more efficient on the larger problems.

Formulation (u_i, u_e)*:* we first consider solving the system (2.4) with a selection of factorized preconditioners, reported in Table 5.2, where D_0 is given by

$$D_0 = C_t + A_e - C_t(C_t + A_i)^{-1}C_t. \tag{5.2}$$

We stress that the use of D_0 is only of theoretical interest, because in general it is too expensive to compute explicitly.

As anticipated in Remark 4.7, other AMG alternatives are explored. The proposed variants only differ for the choice of the matrix block D. Among them and with the aim of simplifying the construction of an approximation to the Schur complement, we first write the matrix $D = C_t + A_e + C_t(C_t + A_i)^{-1}C_t$ as

$$\begin{aligned} D &= [-(C_t + A_e)C_t^{-1}(C_t + A_i) + C_t](C_t + A_i)^{-1}C_t \\ &= [-A_i - A_e - A_eC_t^{-1}A_i](C_t + A_i)^{-1}C_t. \end{aligned}$$

Then only the matrix in brackets is approximated by AMG, yielding

$$\tilde{D} = AMG([-C_t - A_i - A_e + A_e\text{diag}(C_t)^{-1}A_i + C_t])(C_t + A_i)^{-1}C_t. \tag{5.3}$$

Applying $\tilde{D}^{-1}$ entails performing one cycle of Algebraic Multigrid, one multiplication with $C_t + A_i$ and one solve with C_t. This turned out to be more efficient than other approximations such as $D \approx C_t + A_e + C_t\text{diag}(C_t + A_i)^{-1}C_t$.

Prec.	K	D	comments
$\mathcal{P}_f$	AMG$(C_t + A_i)$	AMG$(C_t + A_e)$	cf. Prop. 4.2
$\mathcal{P}_f$	AMG$(C_t + A_i)$	AMG(D_0)	D_0 as in (5.2)
$\mathcal{P}_f$	AMG$(C_t + A_i)$	AMG$(\tilde{D})$	$\tilde{D}$ as in (5.3)
$\mathcal{P}_f$	AMG$(C_t + A_i)$	AMG(A_e)	simplified version

TABLE 5.2

Hence we compare them with the following alternative strategies, where W in the last row is now defined as

$$W = (C_t + A_e) + C_t\text{diag}(C_t + A_i)^{-1}C_t. \tag{5.4}$$

Prec.	K	D	other	comments
$\mathcal{P}_d$	AMG$(C_t + A_i)$	AMG$(C_t + A_e)$		cf. Prop. 4.1
$\mathcal{P}_{whole}$			AMG$(\mathcal{A})$	$\mathcal{A}$ as in (2.4)
$\mathcal{P}_d^{(ic)}$	IC$(C_t + A_i, 10^{-2})$	IC$(W, 3 \cdot 10^{-3})$		W as in (5.4)

We first discuss the numerical experiments associated with $\mathcal{P}_f$.

From the numbers reported in Table 5.3 we readily see that the use of AMG(D_0) for the D matrix soon becomes unacceptably expensive. The same can be said about the use of AMG$(\tilde{D})$, although for the smaller problems this choice is still feasible. Preconditioning with AMG(A_e) represents the most competitive alternative among all the ones we have tried.

$\mathcal{P}_f$, D : n	AMG($C_t + A_e$)	AMG(D_0)	AMG($\tilde{D}$)	AMG(A_e)
2705	3.72 (77)	9.37 (4)	1.21 (14)	0.28 (6)
10657	16.15 (79)	107.29 (5)	12.42 (29)	2.16 (11)
42305	68.14 (80)	–	71.91 (36)	12.95 (15)
168577	312.17 (83)	–	1550.95 (123)	55.82 (15)
673025	1405.74 (86)	–	–	211.37 (13)

TABLE 5.3

(u_i, u_e) formulation. CPU time and number of iterations (in parenthesis) for various variants of the block factorized preconditioner $\mathcal{P}_f$; cf. Table 5.2. "–" stands for excessive CPU time relative to the other approaches.

n	P_d	$\mathcal{P}_{whole}$	$\mathcal{P}_d^{(ic)}$
2705	4.49 (138)	0.14 (3)	0.72 (76)
10657	18.85 (143)	0.71 (3)	2.69 (78)
42305	80.57 (145)	4.33 (4)	14.74 (92)
168577	371.86 (147)	15.66 (4)	78.71 (111)
673025	1649.61 (150)	81.35 (5)	541.23 (165)

TABLE 5.4

(u_i, u_e) formulation: CPU time and number of iterations (in parenthesis) for P_d, $\mathcal{P}_{whole}$ and $\mathcal{P}_d^{(ic)}$.

Table 5.4 reports the CPU times and numbers of iterations for all remaining methods. It is remarkable that the results are significantly different from those obtained for the (u, v) formulation. In particular, AMG preconditioning on the whole matrix $\mathcal{A}$ greatly outperforms all other methods, both in terms of number of iterations and CPU time. Apparently the special structure of the matrix is particularly well suited for the AMG preconditioner we used. In the table below are estimates for the parameters χ_1, χ_2 in the inequalities $\chi_1\mathcal{A} \leq \mathcal{P}_{whole} \leq \chi_2\mathcal{A}$.

n	χ_1	χ_2
2705	1	1.54
10657	1	1.62
42305	1	2.00

Using these estimates to approximate cond($\mathcal{B}$) in (4.1) with $\mathcal{B} = \mathcal{P}_{whole}^{-1}\mathcal{A}$, we clearly see that the error in the energy norm must be below 10^{-6} after about $j = 6$ iterations on the coarser grids.

Among the other methods, both AMG-based block preconditioners show almost mesh independence. Moreover, the best $\mathcal{P}_f$ from Table 5.3 is far cheaper than $\mathcal{P}_d$ because of the much lower number of iterations performed. Block diagonal with incomplete Cholesky is competitive with respect to the AMG block diagonal preconditioner, for the selected choices of K and D, although it clearly shows mesh dependence in the number of iterations.

5.1. Overall computational costs in typical simulations. The setup time for generating the AMG preconditioner usually represents a significant portion of the total cost of this approach. Fortunately, in our case, the same preconditioner can be employed for many time steps, thus fully compensating the initial setup effort. We should add that the AMG preconditioner was used as a black-box within the PIFISS code, and no attempt was made to tune the parameters. We believe that this part of

n	2705		10657		42305	
	$\mathcal{P}_f$ $D = \text{AMG}(A_e)$	$\mathcal{P}_{whole}$	$\mathcal{P}_f$ $D = \text{AMG}(A_e)$	$\mathcal{P}_{whole}$	$\mathcal{P}_f$ $D = \text{AMG}(A_e)$	$\mathcal{P}_{whole}$
nlevel	7-9	10	10-10	10	13-11	12
C_G	2.01-1.73	1.85	2.01-1.74	1.90	1.86-1.74	1.89
C_A	2.65-2.66	2.04	3.43-2.75	3.06	3.07-2.76	3.30
C_S	10.01-12.47	14.78	14.03-15.81	26.85	14.36-18.51	31.63
$C_S^{(1)}$	6.87	13.75	6.93	13.87	6.96	13.93

TABLE 5.5

(u_i, u_e) formulation: number of coarse levels (nlevel), grid complexity C_G, operator complexity C_A, average stencil size across all coarse levels C_S and stencil size of the original matrix $C_S^{(1)}$ for $\mathcal{P}_f$, $D = AMG(A_e)$ and $\mathcal{P}_{whole}$.

the computation could be significantly reduced, if the AMG code were accessible.

In the following we briefly recall the major costs and memory requirements of the algorithm, keeping in mind that the true memory requirements of AMG are not fully reflected by these quantities (some extra work space still needs to be allocated); see [31].

The coarsening strategy and the size of coarse level operators can be deduced from Tables 5.5-5.6 where we report the *grid complexity* C_G, the *operator complexity* C_A and the *average stencil size* C_S defined as:

$$C_G = \left(\sum_{\ell=1}^{nlevel} n_\ell \right) / n_1 \qquad C_A = \left(\sum_{\ell=1}^{nlevel} nnz(T_\ell) \right) / nnz(T_1) \tag{5.5}$$

$$C_S = \frac{1}{nlevel} \sum_{\ell=1}^{nlevel} \frac{nnz(T_\ell)}{n_\ell} \tag{5.6}$$

where *nlevel* denotes the number of levels in the AMG coarsening, T_ℓ is the coefficient matrix at the ℓ-th level (with $T_1 = A$), n_ℓ is the order of T_ℓ and $nnz(T_\ell)$ is the number of non-zeros entries in T_ℓ. The *average stencil size* C_S has to be compared with the average stencil size $C_S(1)$ of the original matrix [31].

The computational cost of applying AMG to a given matrix depends on many factors, including the density of the coefficient matrices and the number of chosen interpolation points, at all levels. C_A provides an indication of the AMG storage requirements, relative to those needed for the original matrix. The ratio C_A/C_G provides a measure of how much denser the matrices T_ℓ's are, compared to the ideal case (the larger the ratio, the denser the matrices). The cost of applying an AMG preconditioner is proportional to the number *nlevel* of coarse levels and the size of operator complexity C_A.

In Tables 5.5-5.6 we report C_A, C_G, C_S and $C_S(1)$ for $\mathcal{P}_f$, $D = AMG(A_e)$ and $\mathcal{P}_{whole}$ with formulation (u_i, u_e), and $\mathcal{P}_f$ with formulation (u_e, v), respectively. Values of C_A in the tables show that memory requirements of AMG based preconditioners amount to three times the one required by the original matrix, and this is in agreement with standard values of AMG complexity [31], including the fact that C_A is basically independent of the matrix size. These fact are strictly related to the simple geometry we are dealing with. Finally, we should remark that Incomplete Cholesky preconditioners are the most efficient in this regard, introducing the least memory overhead of about 70% of the memory requirements of the full original matrix.

n	2705	10657	42305
	$\mathcal{P}_f$	$\mathcal{P}_f$	$\mathcal{P}_f$
$nlevel$	7-9	10-10	13-12
C_G	2.01-1.85	2.01-1.84	1.86-1.84
C_A	2.65-2.95	3.43-3.00	3.07-3.05
C_S	10.01-13.82	14.03-16.01	14.36-18.62
$C_S^{(1)}$	6.87	6.93	6.96

TABLE 5.6

(u_e, v) formulation: number of coarse levels (nlevel), grid complexity C_G, operator complexity C_A, average stencil size across all coarse levels C_S and stencil size of the original matrix $C_S^{(1)}$ for $\mathcal{P}_f$.

n	$\mathcal{P}_d$	$\mathcal{P}_f$	$\mathcal{P}_d^{(ic)}$
2705	2.93/362.93	3.82/593.82	0.22/150.22
10657	12.45/2962.45	14.47/2004.47	0.58/550.58
42305	73/10443	109/13839	2.75/4152.75
168577	1611/53281	1490/44330	12.48/45832
673025	34169/266329	26548/193648	56.25/549296

TABLE 5.7

(u_e, v) formulation: setup times and total execution times for 40 msec (i.e. 1000 time steps) for $\mathcal{P}_d$, $\mathcal{P}_f$ and $\mathcal{P}_d^{(ic)}$.

To evaluate the role of the setup phase, we next report on the results of a longer simulation of the myocardium excitation process. In Tables 5.7-5.8 we display the setup times and the total execution times for a simulation of $40\,msec$, i.e. 1000 time steps, with both formulations. In each case, we only show the best performing preconditioners. For comparison purposes, we also report results with $P_d^{(ic)}$.

AMG preconditioners become more efficient than $\mathcal{P}_d^{(ic)}$ starting from $n = 10657$ and $n = 168577$ for the (u_i, u_e) and (u_e, v) formulation respectively. For $n = 673025$ and the (u_e, v) formulation, we observe a CPU time reduction of 65% and 52% of $\mathcal{P}_f$ and $\mathcal{P}_d$ when compared with $\mathcal{P}_d^{(ic)}$. However, the fastest time is obtained for $\mathcal{P}_{whole}$ with formulation (u_i, u_e). Indeed, a further 30% CPU time reduction can be achieved when compared with $\mathcal{P}_f$ in the (u_e, v) formulation.

The setup phase of AMG based preconditioners can be costly if compared to $P_d^{(ic)}$. Again, we recall that we used PIFISS as a black-box without tuning any parameters and we believe that these setup times could be improved. However, when solving the time–dependent bidomain equations, it is the cost per iteration that determines the efficiency of a method. Thus, memory requirements are the only drawback of AMG based preconditioners, if compared with standard Incomplete Cholesky, but their effectiveness overcomes it.

5.2. The case of discontinuous coefficients. We next report on experiments where the elliptic operators have discontinuous coefficients. We consider new anisotropic conductivity tensors $\tilde{M}_{i,e}$ defined as:

$$\tilde{M}_{i,e} = \begin{cases} \varepsilon M_{i,e} & \forall x \in \Omega_* \\ M_{i,e} & \forall x \in \Omega \setminus \Omega_* \end{cases} \tag{5.7}$$

n	$\mathcal{P}_f$ $D = \text{AMG}(A_e)$	$\mathcal{P}_{whole}$	$\mathcal{P}_d^{(ic)}$
2705	2.34/282.34	3.45/193.43	0.73/110.73
10657	22.53/2182.58	23.05/1433.05	0.41/3580
42305	132.62/13082	145.25/6965	2.18/14952
168577	1445.87/57265	1888.54/17548	12.73/78312
673025	25690.34/237060	53380/134730	62.07/603362

TABLE 5.8

(u_i, u_e) formulation: setup times and total execution times for 40 msec (i.e. 1000 time steps) for $\mathcal{P}_f$, $D = AMG(A_e)$, $\mathcal{P}_{whole}$ and $\mathcal{P}_d^{(ic)}$.

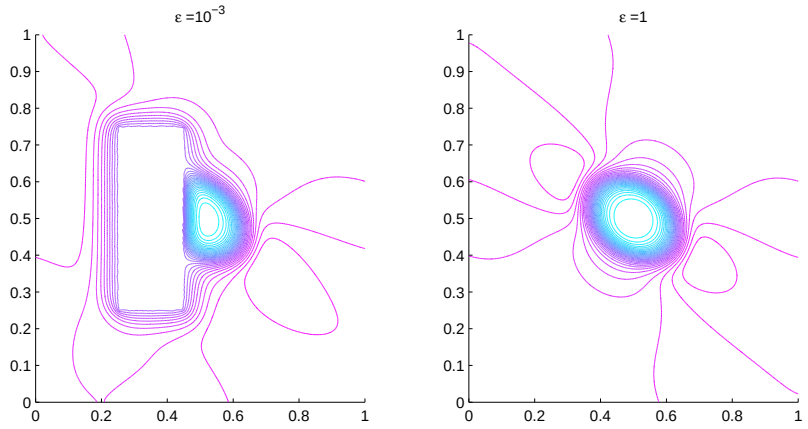

FIG. 5.2. *Extracellular potential maps for discontinuous conductivity coefficients with $\varepsilon = 10^{-3}$ (left) and without discontinuities (right).*

where $\Omega_* \subset \Omega$. Setting $0 < \varepsilon \ll 1$ in Ω_* corresponds to identifying a zone of low or high conductivity in the domain Ω modeling the cardiac tissue.

To test our preconditioners we choose a jump zone $\Omega_* = [0.25, 0.45] \times [0.25, 0.75]$ and the conductivity coefficient $\varepsilon \in [10^{-3}, 10^3]$. Since the coefficients $\sigma_{l,t}^{i,e}$ in the definition of the tensors $M_{i,e}$ are of the order of 10^{-3}, a multiplication by $\varepsilon = 10^{-3}$ simulates regions of no conduction, as displayed in Figure 5.2.

For a typical instant time, we show in Table 5.9 (formulation (u_i, u_e)) and Table 5.10 (formulation (u_e, v)) the number of CG iterations to obtain a reduction of the residual norm by a factor of 10^{-6}, as a function of n and ε. When $\varepsilon = 1$ the anisotropic coefficients $\sigma_{l,t}^{i,e}$ have a uniform distribution and this corresponds to the minimum number of iterations. In both cases, the method seems to be rather insensitive to large jumps in the coefficients, with the number of iterations remaining basically constant. On the other hand, the iteration count of CG with $\mathcal{P}_d^{ic}$ preconditioning is considerably altered by high coefficient discontinuities; cf. Table 5.11 for the (u_i, u_e) formulation.

6. Conclusions. We have presented an experimental comparison of two classes of structured preconditioners for solving the linear systems arising in the numerical solution of two known formulations of a mathematical model in electrocardiology. We have focused our attention on the use of algebraic multigrid based preconditioners, and

ε \ n	2705	10657	42305	168577	673025
10^{-3}	4	4	5	6	6
10^{-2}	4	4	5	6	6
10^{-1}	3	4	5	5	6
1	3	3	4	4	5
10^{1}	5	5	4	4	4
10^{2}	19	11	6	7	5
10^{3}	10	6	3	4	5

TABLE 5.9

(u_i, u_e) formulation: number of PCG iterations for preconditioner $\mathcal{P}_{whole}$ for different values of ε and n.

ε \ n	2705	10657	42305	168577	673025
10^{-3}	8	9	10	11	12
10^{-2}	7	8	9	10	11
10^{-1}	7	8	9	10	11
1	5	7	7	10	10
10^{1}	5	6	7	7	9
10^{2}	5	6	6	7	9
10^{3}	6	6	7	7	9

TABLE 5.10

(u_e, v) formulation: number of iterations for different values of ε and n obtained using $\mathcal{P}_f$.

shown that their effectiveness may be significantly different on the two formulations. In particular, the system stemming from the less exercised (u_i, u_e) formulation may be very efficiently solved by CG with AMG preconditioning on the whole matrix, with no attention to the block structure of the coefficient matrix. This approach largely outperforms all other explored strategies in both formulations when a fine grid is employed. We are planning to generalize the analysis to the crucial three-dimensional case, where AMG preconditioners may become more expensive.

Finally, we have experimentally shown that the analyzed AMG preconditioners are robust with respect to discontinuities in the operator coefficients.

Acknowledgments. We thank Michele Benzi for pointing to relevant references on Algebraic Multigrid preconditioning.

REFERENCES

[1] T. M. Austin, M. L. Trew, and A. Pullan. Solving the cardiac bidomain equations for discontinuous coefficients. *IEEE Trans. Biomed. Eng.*, 53(7):1265–1272, 2006.

[2] O. Axelsson. *Iterative Solution Methods.* Cambridge University Press, New York, 1994.

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

[4] E. Cherry, H. Greenside, and C. S. Henriquez. Efficient simulation of three-dimensional anisotropic cardiac tissue using an adaptive mesh refinement method. *Chaos*, 3, 2003.

[5] I. R Efimov, V. P. Nikolski, and G. Salama. Optical imaging of the heart. *Circ. Res.*, 95(1):21–33, 2004.

[6] P. Colli Franzone, P. Deuflhard, B. Erdmann, J. Lang, and L. F. Pavarino. Adaptivity in

ε \ n	2705	10657	42305	168577	673025
10^{-3}	160	198	219	240	366
10^{-2}	116	142	167	207	316
10^{-1}	82	94	138	169	253
1	76	78	92	111	165
10^{1}	99	125	132	145	201
10^{2}	107	112	126	218	326
10^{3}	441	337	288	277	252

TABLE 5.11
(u_i, u_e) formulation: number of iterations for different values of ε and n obtained using $\mathcal{P}_d^{ic}$.

space and time for reaction-diffusion systems in electrocardiology. *SIAM J. Sci. Comput.*, 28(3):942–962, 2006.

[7] P. Colli Franzone and L. Guerri. Spreading of excitation in 3-D models of the anisotropic cardiac tissue. I: Validation of the eikonal model. *Math. Biosci.*, 113:145–209, 1993.

[8] P. Colli Franzone and L.F. Pavarino. A parallel solver for reaction-diffusion systems in computational electrocardiology. *Math. Models Methods Appl. Sci.*, 14(6):883–911, 2004.

[9] P. Colli Franzone and G. Savaré. *Evolution equations, Semigroups and Functional Analysis, Progr. Nonlinear Differential Equations Appl.*, volume 50, chapter Degenerate evolution systems modeling the cardiac electric field at micro and macroscopic level., pages 49–78. Birkhauser, Basel, 2002.

[10] C. S. Henriquez. Simulating the electrical behavior of cardiac tissue using the bidomain model. *Crit. Rev. Biomed. Engr*, 21(1):1–77, 1993.

[11] N. Hooke, C. S. Henriquez, P. Lanzkron, and D. Rose. Linear algebraic transformations of the bidomain equations: implications for numerical methods. *Math. Biosci.*, 120:127–145, 1994.

[12] G Lines, P Grøttum, and A Tveito. Modeling the electrical activity of the heart –a bidomain model of the ventricles embedded in a torso. *Computing and Visualization in Science*, 5(4):195–213, 2003.

[13] C. Luo and Y. Rudy. A model of ventricular cardiac action potential: depolarization, repolarization, and their interaction. *Circ. Res.*, 68:1501–1526, 1991.

[14] The MathWorks, Inc., Natick, Mass. 01760. *MATLAB User's Guide*, 2008.

[15] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra.* SIAM, Philadelphia, 2000.

[16] M. Murillo and X. C. Cai. A fully implicit parallel algorithm for simulating the non-linear electrical activity of the heart. *Numer. Linear Algebr. Appl.*, 11:261–277, 2004.

[17] Y. Notay. Algebraic multigrid and algebraic multilevel methods: a theoretical comparison. *Numer. Linear Algebr. Appl.*, 12:419–451, 2005.

[18] M. Pennacchio. The mortar finite element method for the cardiac "bidomain" model of extracellular potential. *J. Sci. Comput.*, 20:191–210, 2004.

[19] M. Pennacchio, G. Savaré, and P. Colli Franzone. Multiscale modelling for the bioelectric activity of the heart. *SIAM J. Math. Anal.,*, 37:1333–1370, 2006.

[20] M. Pennacchio and V. Simoncini. Efficient algebraic solution of reaction-diffusion systems for the cardiac excitation process. *J. Comput. Appl. Math.*, 145(1):49–70, 2002.

[21] M. Pennacchio and V. Simoncini. Substructuring preconditioners for mortar discretization of a degenerate evolution problem. *J. Sci. Comput.*, 36:391–419, 2008.

[22] G. Plank, M. Liebmann, R. Weber dos Santos, E. J. Vigmond, and G. Haase. Algebraic multigrid preconditioner for the cardiac bidomain model. *IEEE Trans. Biomed. Eng.*, 54:585–596, 2007.

[23] C. E. Powell. *Optimal Preconditioning for Mixed Finite Element Formulations of Second-Order Elliptic Problems.* PhD thesis, UMIST, 2003.

[24] C. E. Powell and D. Silvester. Optimal preconditioning for Raviart-Thomas mixed formulation of second-order elliptic problems. *SIAM J. Matrix Anal. Appl.*, 25:718–738, 2003.

[25] C. E. Powell and D. Silvester. PIFISS Potential (Incompressible) Flow & Iterative Solution Software Guide. Technical Report 2007.14, MIMS Preprint, 2007.

[26] W. Quan, S. T. Evans, and H. M. Hastings. Efficient integration of a realistic two–dimensional cardiac tissue model by domain decomposition. *IEEE Trans. Biomed. Eng.*, 45(3):372–385,

1998.

[27] Y. Saad. *Iterative methods for sparse linear systems*. The PWS Publishing Company, 1996.

[28] H. I. Saleheen and K. T. Ng. A new three–dimensional finite difference bidomain formulation for the inhomogeneous anisotropic cardiac tissues. *IEEE Trans. Biomed. Eng.*, 45(1):15–25, 1998.

[29] K. Skouibine and W. Krassowska. Increasing the computational efficiency of a bidomain model of defibrillation using a time–dependent activating function. *Ann. Biomed. Eng.*, 28:772–780, 2000.

[30] B. F. Smith, P. E. Björstad, and W. D. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[31] K. Stüben. A review of algebraic multigrid. Technical Report GMD-Report 69, German National Research Center for Information Technology (GMD), Institute for Algorithms and Scientific Computing (SCAI), St. Augustin, Germany, 1999.

[32] A. Toselli and O. Widlund. *Domain Decomposition Methods - Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, 2004.

[33] J. Trangestein and C. Kim. Operator splitting and adaptive mesh refinement for the luo–rudy i model. *J. Comput. Phys.*, 196:645–679, 2004.

[34] M. Veneroni. Reaction-diffusion systems for the macroscopic bidomain model of the cardiac electric field. Technical Report 25-PV, IMATI-CNR, Pavia, 2006.

[35] E. J. Vigmond, F. Aguel, and N. A. Trayanova. Computational techniques for solving the bidomain equations in three dimensions. *IEEE Trans. Biomed. Eng.*, 49(11):1260–1269, 2002.

[36] E. J. Vigmond, R. Weber dos Santos, A. J. Prassl, M. Deo, and G. Plank. Solvers for the cardiac bidomain equations. *Progress in Biophysics & Molecular Biology*, 96:3–18, 2008.

[37] R. Weber dos Santos, G. Plank, S. Bauer, and E. J. Vigmond. Parallel multigrid preconditioner for the cardiac bidomain model. *IEEE Trans. Biomed. Eng.*, 51(11):1960–1968, 2004.

[38] R. Weber dos Santos, G. Plank, S. Bauer, and E. J. Vigmond. Preconditioning techniques for the bidomain equations. In *Lecture Notes In Computational Science And Engineering*, volume 40, pages 571–580. Springer, New York, 2004.