ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

MATHICSE

Mathematics Institute of Computational Science and Engineering

School of Basic Sciences - Section of Mathematics

MATHICSE Technical Report

Nr. 07.2016

February 2016

Improved hybrid/GPU algorithm for solving cardiac electro-physiology problems on Purkinje networks

M. Lange, S. Palamara, T. Lassila, C. Vergara, A. Quarteroni, A.F. Frangi

http://mathicse.epfl.ch

Phone: +41 21 69 37648

Address:
EPFL - SB - MATHICSE (Bâtiment MA)
Station 8 - CH-1015 - Lausanne - Switzerland

Fax: +41 21 69 32545

Submitted to INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN BIOMEDICAL ENGINEERING
Int. J. Numer. Meth. Biomed. Engng. 2015; **00**:1–**??**
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cnm

Improved hybrid/GPU algorithm for solving cardiac electrophysiology problems on Purkinje networks

M. Lange[1], S. Palamara[2], T. Lassila[1], C. Vergara[2], A. Quarteroni[3], A.F. Frangi[1]

[1]*CISTIB, Department of Electronic and Electrical Engineering, The University of Sheffield, UK*
[2]*MOX, Dipartimento di Matematica, Politecnico di Milano, Italy*

[3]*CMCS, Mathematics Institute of Computational Science and Engineering, École Polytechnique Fédérale de Lausanne, Switzerland*

SUMMARY

The cardiac Purkinje fibres provide an important stimulus to the coordinated contraction of the heart. We present a numerical algorithm for the solution of electrophysiology problems on the Purkinje network that is efficient enough to be used on realistic networks with physiologically detailed membrane models. The algorithm is based on operator splitting and is provided with three different implementations: pure CPU, hybrid CPU/GPU, and pure GPU. Compared to our previous work based on the model of Vigmond et al., we modify the explicit gap junction term at network bifurcations in order to improve its mathematical consistency. Due to this improved consistency of the model, we are able to perform a convergence study against analytical solutions and verify that all three implementations produce equivalent convergence rates. Finally, we compare the efficiency of all three implementations on Purkinje networks of increasing spatial resolution using membrane models of increasing complexity. Both hybrid and pure-GPU implementations outperform the pure-CPU implementation, but their relative performance difference depends on the size of the Purkinje network and the complexity of the membrane model used. Copyright © 2015 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: electrophysiology; Purkinje networks; graphics processing units; hybrid algorithms

1. INTRODUCTION

The Purkinje fibres form an extensively branching network of fast conducting cells within the ventricular sub-endocardium. This network covers large areas of the ventricles and initiates their rhythmic contraction in response to signals traversing from the atrio-ventricular (AV) node along the His bundle and its branches [**?**, **?**]. The fibres consists of Purkinje cells with ion channels allowing ion exchange between the intra- and extra-cellular compartments when a threshold in transmembrane potential difference is reached, triggering a traveling electrical potential wave (the "action potential") that propagates along the Purkinje fibre and finally enters the ventricular myocardium at the Purkinje-muscle junctions (PMJs). Since the action potentials in the Purkinje fibres are quite different from those of the myocardial cells, the ability to simulate propagation of action potentials based on these ion channels in physiological Purkinje networks is essential for studies of the whole heart to obtain realistic activation patterns. In pathological hearts disturbances

*Correspondence to: Centre for Computational Imaging & Simulation Technologies in Biomedicine (CISTIB), Department of Electronic and Electrical Engineering University of Sheffield, Pam Liversidge Building, Mappin Street, Sheffield S1 3JD, UK.

in the conduction system can alter the activation pattern greatly, e.g. through bundle branch blocks, and give rise to different types of arrhythmias due to re-entry into the Purkinje network or ectopic beats arising from Purkinje fibre automaticity [**?**, **?**, **?**, **?**, **?**, **?**]. Recent computational studies have also shown that neglecting to model the Purkinje system at sufficient levels of detail can lead to ventricular activation patterns that do not correspond to physiological ones [**?**, **?**].

While the importance of Purkinje-myocardium interaction is established in cardiology, especially in the study of arrhythmias, relatively few computational heart models incorporate activation in a detailed Purkinje network. Many works focusing on detailed whole-heart myocardium modelling only consider the positions of the PMJs as stimulus currents in the myocardium model, or prescribe unrealistic "apex-only" or "full-endocardium" initial activation patterns. To simulate the propagation of an action potential in a Purkinje network the bidomain equation [**?**], the cable equation with a reaction term [**?**], and the Eikonal equation [**?**] have been used. The latter model is the most basic one, modelling only the propagation delay from the AV node to the PMJs in order to obtain the initial myocardial pattern and then proceeding with solving action potentials only in the myocardium. However, recent computational studies have shown that the Eikonal approximation is unsatisfactory for the Purkinje network, due to the existence of a "push-and-pull" -effect at the branching points [**?**]. Therefore, a more detailed model is necessary. The approach of Vigmond and Clements [**?**] uses the monodomain equation to simulate the conduction in a tree of branching Purkinje fibres. All of these models currently lack experimental validation due to difficulties in measuring action potentials in the subendocardial Purkinje cells. As such, the importance of numerical verification first and foremost has been identified as a key stepping stone to the wider development and acceptance of numerical models and methods for simulating Purkinje network activation [**?**].

We extend our previously developed implementation [**?**] of the explicit gap junction model of Vigmond and Clements [**?**] by modifying the gap junction terms at network bifurcation points from their original formulation. This proves to be necessary for the consistency of the method at the limit of $h \to 0$, so that the convergence rate of the method can be verified. The algorithmic implementations come in three different versions: CPU only, CPU/GPU hybrid, and GPU only. In general, a GPU can be used to accelerate the simulations due to the embarrassingly parallel nature of the membrane model evaluations, as has been shown before for the monodomain equation in [**?**, **?**]. To ensure that all our implementations produce identical and reliable results, we develop a verification test for the Purkinje network electrophysiology solver.

Our method of verification is the same as typically used for numerical solvers: the difference between an analytical solution and the numerical solution is computed. Once the solution accuracy has been verified, a second comparison is made on the performance of the different solvers for the same complex problem [**?**, **?**]. Analytical solutions for the cable equation with simplified membrane models are known in equilibrium [**?**, **?**] and for travelling waves [**?**, **?**, **?**], but in all cases the solutions are developed on a line without branching points. To verify the correct behaviour at the branching line segments, we develop equilibriums solutions of the cable equation with a simplified membrane model on a branching structure of 1-D line segments. The temporal dependency of the solution, i.e. propagation speed and wave front behaviour, are verified on a line segment without branching.

Finally, we evaluate the performance of the three implementations by simulating activation on four different Purkinje fibre networks and two ionic membrane models of increasing complexity. The Purkinje networks vary in number of branches, while the complexity of the membrane models varies in the number of equations. The performance of the different implementations can then be evaluated with respect to the computational time spent in each of the different configurations. The developed efficient and parallel GPU-based implementation of the solver turns out to be especially attractive in conjunction with detailed membrane models for human Purkinje-cells at the problem sizes required to simulate realistic whole-heart activation patterns.

DOI: 10.1002/cnm

2. NUMERICAL SCHEME FOR COMPUTING ACTION POTENTIALS IN 1-D NETWORKS

2.1. *Definition and modification of the explicit gap junction model*

To solve the action potential in the Purkinje fibres, we improve our previously published algorithm [**?**], which is described below. It is based on the one-dimensional *monodomain equation*

$$\partial_x \left(\sigma_i \partial_x V_m\right) = \beta(C_m \partial_t V_m + I_{\text{ion}}(V_m, \xi)), \tag{1}$$

where V_m is the transmembrane potential, I_{ion} the total current through the ionic channels on the cell membrane, ξ are the state variables of the membrane model, β is the surface-to-volume ratio of the cell membrane, C_m the cell membrane capacitance, σ_i the intracellular conductivity tensor, and x the local spatial coordinate.

The one-dimensional cable equation needs to be extended to describe the propagation of the potential at the branching points. Each Purkinje branch is modelled as a separate problem on a one-dimensional line segment, which is then coupled to other branches by interface conditions determined by the continuity of potential and Kirchhoff's current law. For the latter, the input and output currents at the branching points are needed, which can either be obtained from a numerical derivative of the potential, or by the use of a finite difference approximation for the derivative. The numerical problem in this work is solved with a finite element (FE) scheme using Hermite basis functions, such that the derivatives of the solutions are degrees of freedom (DOF) in the formulation.

To close the monodomain equations approximated using Hermite basis functions, the concept of explicit *gap junction* models is introduced. Gap junctions are specialised intercellular connections between two or more Purkinje cells at their ends (see Fig. 1). In the classical monodomain model for the myocardium, the effect of these gap junctions is hidden in the conductivity tensor by a formal homogenisation process [**?**], whereas in the Purkinje network the gap junctions are explicitly modelled [**?**, **?**, **?**] in order to correctly capture the "push-and-pull" -effect.

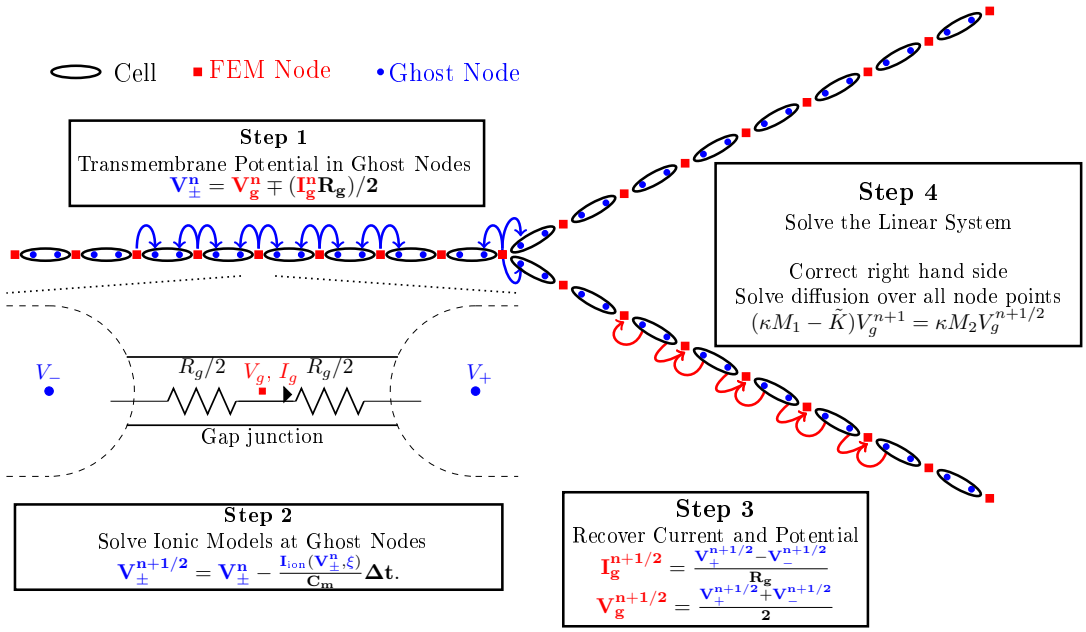

Figure 1. Illustration of three Purkinje branches and the gap junctions that links together the Purkinje cells. The four steps of the algorithm to solve electrophysiological problems are shown. In the middle of the gap junction are the intra cellular potential V_g and the current I_g. Additional nodes (*ghost nodes*) are defined to compute the cell membrane model in the cell. All equations are explained in the text.

Let us consider the case of two Purkinje cells connected by a gap junction (inset of Fig. 1). Then the DOFs of the problem are the intracellular potential V_g and the current I_g across the gap junction,

DOI: 10.1002/cnm

which are formally located in the middle of the gap junction (in red in Fig. 1). The ionic channel current I_{ion} is calculated in the cells at the ghost nodes (in blue in Fig. 1), which means for each node point the cell membrane model needs to be evaluated twice.

The relation between the intracellular potential ϕ_i and the transmembrane potential in the cells $V_\pm$ is given by Ohm's law

$$V_\pm = \phi_i - \phi_e \mp \frac{I_g R_g}{2},$$

where R_g is the gap junction resistance and ϕ_e is the extracellular potential. The latter is assumed to be constant in this work. Furthermore, up to a multiplicative factor I_g represents the derivative of ϕ_i by Ohm's law. The difference between intra- and extra cellular potential is the transmembrane potential $V_g = \phi_i - \phi_e$.

The values ϕ_i, I_g at the branching point are repeated in order to allow each segment to be solved separately. The three endpoints of the segments are then assumed to connect in the gap junction of the three cells (see Fig. 2). In contrast to our previously published method [**?**], each of the points gets only one cell membrane model associated with it instead of two. The single cell membrane model will then be solved in the corresponding cell segment. The currents are given from each cell to the branching point as indicated in Fig. 2. This adjustment is necessary because in the previous work there are six ghost nodes in each branching point, but only three actual cells. To get a more accurate comparison with analytical solutions we have modified our method accordingly.

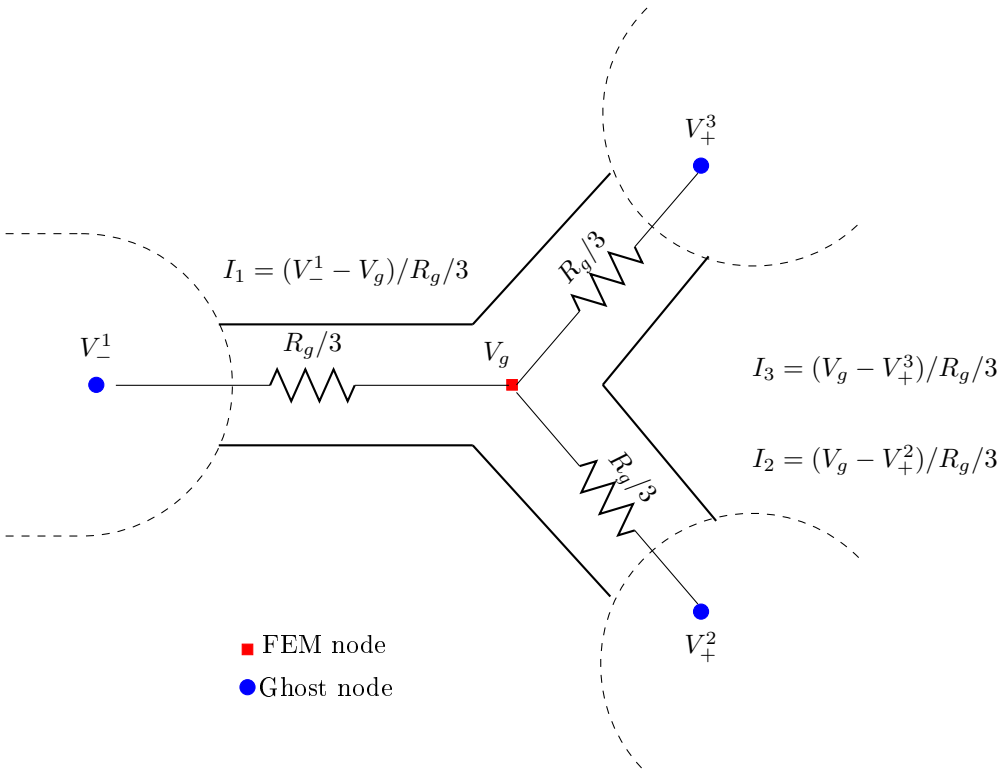

Figure 2. Detail of the branching node point in the finite element (FE) node points and ghost nodes. The currents $I_{1,2,3}$ are defined from the ghost node the FE nodes.

Once the explicit gap junction model is in place, we need to ensure the compatibility of the macroscopic conductivity tensor with the gap junction resistance. Therefore, we assume that σ_i is the intracellular conductivity without any effect of gap junctions and introduce the equivalent conductivity under the assumption of a cylindric volume conductor $\sigma_i^* = (\sigma_i l)/(\ell + \sigma_i R\pi\rho^2)$, where ℓ is the length of the Purkinje cell and ρ its radius. Note that this assumes that any discretisation has a step length $h = Z\ell$, which is an integer multiple Z of the cell length ℓ. In this

DOI: 10.1002/cnm

notation (1) becomes

$$\partial_x \sigma_i^* \partial_x V_\pm = \beta(C_m \partial_t V_\pm + I_{\text{ion}}(V_\pm, \xi_\pm)). \tag{2}$$

To approximate the solution of (2) in time we introduce a time discretisation, where the superscript n refers to the numerical solution computed at time t^n. The algorithm to obtain the solution at t^{n+1} from the solution at t^n has four steps (Fig. 1), and uses of the operator splitting scheme as follows

$$\begin{cases} \partial_t V + L_1(V) = 0 \\ \partial_t V + L_2(V) = 0 \end{cases}, \tag{3}$$

where $L_1 = I_{\text{ion}}$, $L_2 = \partial_x (\sigma_i^* \partial_x)$ and V being one of the unknown potentials. The first three steps address the first equation, starting with Ohm's law to obtain the transmembrane potential $V_\pm^n$ in the cells from the potential V_g^n and current I_g^n at the gap junctions

$$V_\pm^n = (\phi_i^n - \phi_e^n) \mp \frac{I_g^n R_g}{2}.$$

The algorithm then proceeds to the second step, in which the membrane models are solved

$$V_\pm^{n+1/2} = V_\pm^n - \frac{I_{ion}(V_\pm^n, \xi)}{C_m} \Delta t.$$

The third step generates from the transmembrane potentials $V_\pm^{n+1/2}$ the current and potential in the gap junction $I_g^{n+1/2}, V_g^{n+1/2}$. The last step solves the operator equation $\partial_t V + L_2(V) = 0$ with the FE model, therefore we use the relation

$$V_g = \phi_i^{n+1/2} - \phi_e^{n+1/2} = \frac{V_+^{n+1/2} + V_-^{n+1/2}}{2}$$

and since the equations (2) are linear in $V_\pm$ we obtain:

$$\beta C_m \frac{(\phi_i^{n+1} - \phi_e^{n+1}) - (\phi_i^{n+1/2} - \phi_e^n)}{\Delta t} = \partial_x \sigma_i \partial_x \phi_i^{n+1}. \tag{4}$$

In the FEM this translates into the following matrix formulation of the problem:

$$(\kappa M - K)\tilde{\phi}_i^{n+1} = \kappa M(\tilde{\phi}_i^n + (\tilde{\phi}_e^{n+1} - \tilde{\phi}_e^n)), \tag{5}$$

where $\kappa = \beta C_m / \Delta t$, M is the mass matrix, K is the stiffness matrix and $\tilde{\phi}_i$ is the vector of unknowns, which contains the potential and the derivative of the potential at each node. This is due to the fact that we are using Hermite basis functions, which include as a DOF also the value of the derivative in each node.

In the stiffness matrix of the FE model, the triplicated branching points of line segments are used to enforce the interface conditions and thus couple together the solutions obtained from the different line segments. In the case that segment 1 branches into segments 2 and 3, we enforce the continuity of the potential $\phi_1 = \phi_2 = \phi_3$ and the conservation of current $I_1 = I_2 + I_3$. In contrast to Vigmond and Clements [**?**], our implementation covers the case where segments 1 and 2 join to form segment 3 and thus giving our algorithm the ability to solve physiological networks with loops. In this case the coupling condition of the currents is $I_1 = I_3 - I_2$. These boundary conditions are introduced in the FEM stiffness matrix associated to (5) and the right hand side.

2.2. Hardware implementation

In the following section, we detail the different characteristics of the three implementations. They are all performed using the `LifeV` library[†], which provides methods to assemble the FE stiffness

[†]The `LifeV` (http://www.lifev.org) finite element library is the joint collaboration between four institutions: EPFL, Politecnico di Milano, INRIA, and Emory University.

DOI: 10.1002/cnm

and the mass matrices. Furthermore, linear solvers and preconditioners are provided through the `Trilinos`[‡] linear algebra library.

In all implementations the linear system is solved with the generalised minimal residual method (GMRES) with ILU factorisation for preconditioning of the linear system. The GMRES method was used as the system matrices are not symmetric due to the coupling condition enforcement at the junctions. The pure CPU implementation was parallelised with the help of the OpenMPI framework, which allows in the proposed algorithm to perform Steps 1 to 3 in a distributed way with linear partitioning. The linear system is solved with one OpenMPI process to eliminate communication between CPUs while solving the linear system. Furthermore, the computational most expensive step in the algorithm is Step 2. This implies that all other processes need to send their data to the serial process and after solving the problem the solution needs to be redistributed (see Fig. 3 for the workflow).

In the CPU/GPU hybrid implementation the membrane models are computed with the GPU. Therefore, before Step 2 the transmembrane potential is copied to the GPU, then the membrane model variables are updated, and the transmembrane potential is copied back from the GPU to the CPU. These three tasks are generated and queued in a CUDA streams, which allow for asynchronous GPU tasks. After the CPU has scheduled all task groups, it waits for their completion, and subsequently returns to Steps 3 and 4 on the CPU.

The third implementation does all the computation on the GPU, thus there is no memory copy between the steps. Steps 1 and 3 use the same code on the GPU as on the CPU, and in Step 2 we reuse the code from the hybrid implementation, but without the memory copy. To solve the linear system in Step 4, the mass matrix and the stiffness matrix are built on the CPU with the `LifeV` framework, and the resulting sparse matrices are copied to the GPU. The same is done for the preconditioner, which is built on the CPU and then copied to the GPU. The GMRES method on the GPU is the same as on the CPU, but uses `cuSPARSE` and `cuBLAS` for the matrix operation. In each iteration, the preconditioned linear system is solved with a forward and backward solver. The CUDA framework provides a parallel implementation of this [**?**].

There are two different hybrid and GPU implementations in the performance test, which are due to different precision used. GPUs are designed for single precision and thus have much higher number of floating point operations in single precision than in double precision mode. Therefore, we implemented the same GPU code using both double precision, and selectively dropping down to single precision where numerical stability was verified not to be affected. This is referred to as *mixed-mode*.

All computations were performed with a Dell Precision-WorkStation-T7500 featuring two Intel(R) Xeon(R) CPUs E5620 at 2.40GHz and a NVIDIA Quadro 4000 GPU with 256 CUDA Cores.

3. VERIFICATION OF THE PROPOSED NUMERICAL METHOD

Two verification tests are performed. The first evaluates the accuracy of the solution in equilibrium against an analytical solution, and the second uses a travelling pulse to verify the dynamic solution.

3.1. Numerical error and convergence in equilibrium

In the first experiment we seek an equilibrium solution for the monodomain equation (2), of a caricature model [**?**] given by

$$\partial_t V = pV, \tag{6}$$

where V is the transmembrane potential and p is a model parameter. Depending on p the cells are stable ($p < 0$) and return exponential to 0 from any deflection, or they are unstable ($p > 0$) and the

[‡]http://www.trilinos.org

DOI: 10.1002/cnm

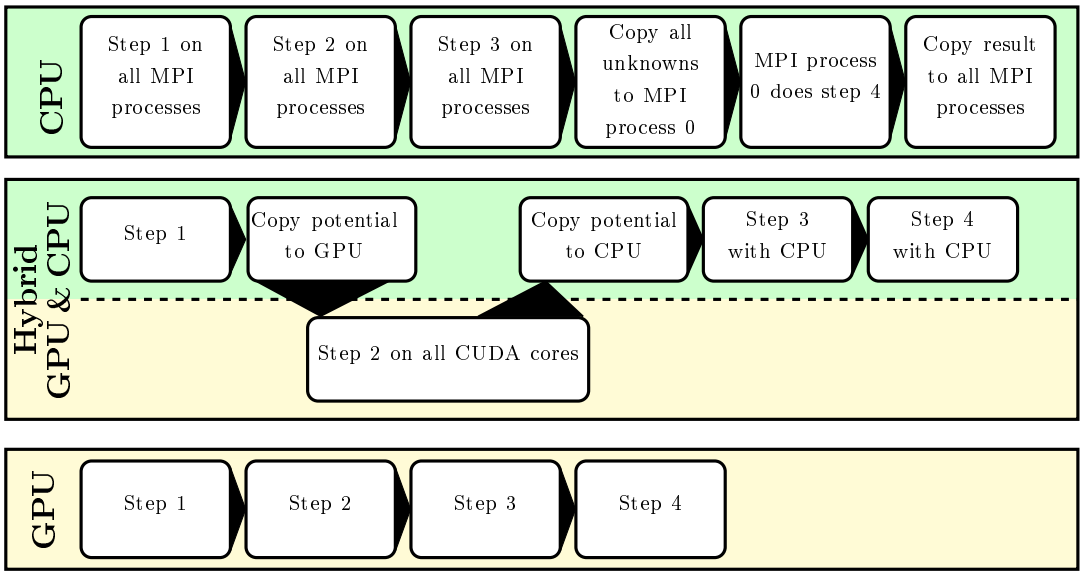

Figure 3. The workflow for the CPU (above), CPU/GPU hybrid (middle), and GPU (lower) implementation. The CPU needs to copy the transmembrane potential in the gap junctions and the current, while the CPU/GPU hybrid needs to copy the potential of the ghost nodes. In the GPU implementation there is no copy required.

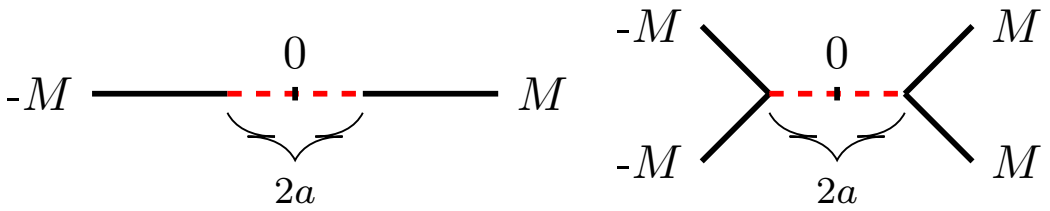

Figure 4. Problem domains for the equilibrium solutions, where the dashed area are unstable cells. Left for a line segment and right for symmetric bifurcation

transmembrane potential increases exponentially. The cell membrane model is then applied to two different test geometries. The first geometry is a finite one-dimensional line segment (Fig. 4, left) $D_1 = [-M, M]$, $M > 1$, which is divided in three subparts $D_{1,1} = [-M, -1]$, $D_{1,2} = [-1, 1]$, and $D_{1,3} = [1, M]$. In $D_{1,1}$ and $D_{1,3}$ the cell membrane model is chosen to be stable while in $D_{1,2}$ it is unstable

$$p(x) = \begin{cases} p_1 & \text{for } x \in D_{1,2} \\ -p_2 & \text{else where} \end{cases},$$

where $p_i > 0$. The simplified cell membrane model is then introduced in the monodomain equation (1). Letting $\delta = \sigma_i/\beta$ and assuming that the conductivity σ_i has no spatial dependency the problem to be solved becomes

$$\begin{aligned} C_m \partial_t V &= \delta \partial_x^2 V - p(x)V, \\ V_1(-1) &= V_2(-1), \quad V_2(1) = V_3(1), \\ V_1'(-1) &= V_2'(-1), \quad V_2'(1) = V_3'(1), \\ V_1(-M) &= 0, \quad V_3(M) = 0. \end{aligned} \tag{7}$$

DOI: 10.1002/cnm

The solution can be deduced with the canonical ansatz $V_i(x) = c_1 \exp(kx) + c_2 \exp(kx)$, as shown by Artebrant *et al.* [**?**]. For a line segment the solution is:

$$V(x) = \begin{cases} \sinh(\kappa(M+x)) & , \quad x \in D_{1,1} \\ d\cos(kx) & , \quad x \in D_{1,2} \\ \sinh(\kappa(M-x)) & , \quad x \in D_{1,3} \end{cases}, \tag{8}$$

where $d = \sinh(\kappa(M-1))/cos(k)$ and parameters $\kappa = \sqrt{p_1/\delta}$ and $k = \sqrt{p_2/\delta}$. To satisfy the differentiability conditions $\mathbf{V_1'}(\mathbf{x})|_{\mathbf{x=-1}} = \mathbf{V_2'}(\mathbf{x})|_{\mathbf{x=-1}}$, $\mathbf{V_2'}(\mathbf{x})|_{\mathbf{x=1}} = \mathbf{V_3'}(\mathbf{x})|_{\mathbf{x=1}}$, in (7) the relation

$$k\tan(k) = \frac{\kappa}{\tanh(\kappa(M-1))}$$

must hold.

The second problem is a symmetric domain D_2 with a branching and joining point (Fig. 4, right). The domain consists of five line segments, the first two, $D_{2,1} = [-M, -1]$ and $D_{2,2} = [-M, -1]$, join the segment $D_{2,3} = [-1, 1]$, which branches into two further segments $D_{2,4} = [1, M]$ and $D_{2,5} = [1, M]$. As in the first domain, the middle segment $D_{2,3}$ has unstable cells while the outer branches $D_{2,1}, D_{2,2}, D_{2,4}$, and $D_{2,5}$ are stable. The problem is symmetric at zero, so we will look at the negative domain only. Furthermore, $D_{2,1}$ and $D_{2,2}$ are equal, thus it is sufficient to find the solution on one of them. This means we need to solve the following problem

$$\begin{array}{rl} \delta V_1'' - p_1 V_1 = 0 & \forall\, x \in D_{2,1} \\ \delta V_3'' + p_2 V_3 = 0 & \forall\, x \in D_{2,3} \\ V_1(-1) = V_3(-1),\ 2V_1'(x)|_{x=-1} = V_3\prime(x)|_{x=-1},\ V_1(-M) = 0 & \end{array},$$

where the first two equations are due to Kirchoff's current law, which states that the current sum of the first and second branch need to equal the third branch.

Following an exponential ansatz, we constrain the solution to be unique by choosing the maximum amplitude $V(0) = 1$, which leads to

$$\begin{aligned} V_{1,2} &= c_1 \sinh(\kappa(M+x)), \\ V_3 &= \cos(kx), \\ V_{4,5} &= c_1 \sinh(\kappa(M-x)), \end{aligned} \tag{9}$$

where $c_1 = \cos(-k)/\sinh(\lambda_1(M-1))$ and the relation between κ and k changes to

$$k\tan(k) = \frac{2\kappa}{\tanh(\kappa(M-1))}.$$

3.1.1. Numerical solution in equilibrium For the numerical solution we choose $\delta = 1$, which imposes the condition $1 = \sigma^*/\beta$. With a physiological cell length of $\ell = 62.5$ μm, diameter 16.0 μm, and chosen gap-junction resistance $R = 0.1$ kΩ, the intracellular conductivity becomes 1967.5 kS/cm. Furthermore, the spatial step size h is chosen as an integer multiples of the cell length ℓ.

On the line D_1 we choose parameters $p_2 = 0.0946441$, $M = 20$, and the capacitance of the cell membrane $C_m = 1$ μF. For the branching domain D_2 we choose $M = 10$, $p_2 = 1$, and $\delta = 1$.

The resulting error distribution over the line and the branching domain is shown in Fig. 5, where the largest contribution of the error comes from the passive cell region. The convergence test shows that with decreasing spatial step size the L_2-error reduces faster than linearly for the single-interval domain D_1 (Fig. 5) and linearly for the example in the branching domain D_2. Note that without the modification introduced in Sect. 2.1 only sub-linear convergence behaviour was obtained for the branching domain case (compare with Fig. 3 from [**?**]), indicating that the modification is required for the consistency of the numerical method.

DOI: 10.1002/cnm

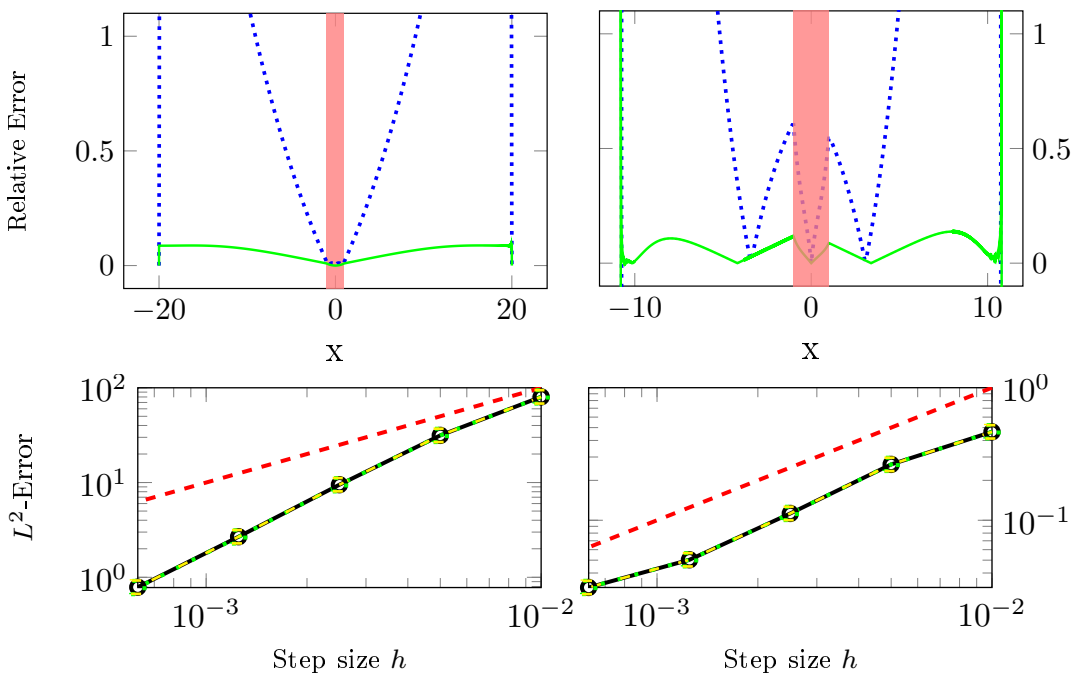

Figure 5. Error obtained with different numerical implementations. Upper row: relative error obtained for the pure-GPU implementation. Solution over one line segment D_1 on the left, over the branching configuration D_2 on the right. Blue dotted line corresponds to error for a step size of 0.1 mm, the green solid line the error (scaled by 10) for step size of 0.00625 mm. The red zone corresponds to the domain of active cells. Lower panel: L_2 error as a function of spatial step size h for different implementations; black solid line for CPU, yellow dashed line for hybrid, green dotted line for GPU, and red dashed line the comparison with linear convergence rate.

3.2. Analytical solution for a travelling pulse

In this section the convergence of the dynamic solution is investigated. Therefore, an analytic solution for a travelling wave of the linearised FitzHugh-Nagumo (FHN) equation [**?**, **?**] is constructed and then solved numerically.

For the construction of the solution to the FHN model we follow the work of Rinzel and Keller [**?**], who used the two variable model, with the transmembrane potential V and the recovery variable w

$$\begin{aligned} \partial_t V = I_{ion} &= f(V) + w \\ \partial_t w &= bV \\ f &= V(\alpha - V)(1 - V) \end{aligned} \quad , \tag{10}$$

where f can be linearised to $f = V - H(V - \alpha)$, with $0 \leq a \leq 1/2$ and H is the Heaviside function. The solution of the linearised problem (10) on an infinite line is well-known [**?**, **?**].

We consider the monodomain equation (2) over an infinite line under the assumption that σ_i^* does not depend on x and couple it to the linearised FHN cell membrane model

$$\begin{aligned} C_m \partial_t V &= \tfrac{\sigma_i^*}{\beta} \partial_x^2 V - f(V) - w \\ \partial_t w &= bV \qquad\qquad\qquad\qquad , b \geq 0 \\ f(V) &= V - H(V - a) \qquad\quad , 0 \leq a \leq 1/2 \end{aligned} \quad . \tag{11}$$

To be in the condition of the approach in [**?**] we assume in the following $\sigma_i^*/\beta = 1$ and $C_m = 1$.

DOI: 10.1002/cnm

By differentiating the first equation in (11) with respect to time, the system can be rewritten in one equation

$$\begin{aligned} \partial_t^2 V &= \partial_t \partial_x^2 V - \partial_t f(V) - \partial_t w, \\ \Rightarrow \partial_t^2 V &= \partial_t \partial_x^2 V - \partial_t f(V) - bV. \end{aligned} \tag{12}$$

To solve this problem the travelling wave ansatz $V(x,t) = v_c(z)$ where $z = x + ct$ with $c > 0$ is introduced. Furthermore, we assume that $v_c(0) = a$ and $\lim_{|z|\to\infty} v_c(z) \to 0$, and from the intermediate value theorem follows the existence of a $z_1 \neq 0$ with $v_c(z_1) = a$. The system to be solved can be rewritten as

$$\begin{aligned} c^2 v_c'' &= c v_c''' - c f'(v_c) v_c' - bv \\ 0 &= v_c''' - c v_c'' - f'(v_c) v_c' - (b/c) v \\ 0 &= \begin{cases} v_c''' - c v_c'' - v_c' - (b/c) v & \forall z \in \mathbb{R} \backslash \{0, z_1\} \\ v_c''' - c v_c'' - (b/c) v & z \in \{0, z_1\} \end{cases}, \end{aligned} \tag{13}$$

with boundary condition $\lim_{|z|\to\infty} v_c(z) \to 0$ and where $'$ indicates a derivative with respect to z. The solution can be obtained in the three regions $z < 0$, $0 \leq z \leq z_1$ and $z > z_1$. Following an exponential ansatz for the differential equation we need to find the roots of the cubic polynomial

$$p(\lambda) = \lambda^3 - \lambda^2 - \lambda - (b/c). \tag{14}$$

If the discriminant is non-negative there are three distinct real solution, while for a negative discriminate two of the solutions are complex. Let λ_1 be the positive real solution while λ_2 and λ_3 are the possible complex. Than the solution to the differential equation (13) is [**?**]

$$v_c = \begin{cases} a \exp(\lambda_1 x) & z < 0 \\ (a - p'(\lambda_1))^{-1} \exp(\lambda_1 x) - p'(\lambda_2)^{-1} \exp(\lambda_2 x) - p'(\lambda_3)^{-1} \exp(\lambda_3 x) & 0 \leq z \leq z_1 \\ p'(\lambda_2)^{-1}(\exp(-\lambda_2 z_1) - 1) \exp(\lambda_2 x) + p'(\lambda_3)^{-1}(\exp(-\lambda_3 z_1) - 1) \exp(\lambda_3 x) & z > z_1 \end{cases}. \tag{15}$$

In the following we show that v_c is real, even with complex eigenvalues λ_2, λ_3. We use that $\mathrm{Re}(\lambda_2) = \mathrm{Re}(\lambda_3)$ and $\mathrm{Im}(\lambda_2) = -\mathrm{Im}(\lambda_3)$, where $i = \sqrt{-1}$.

$$\Rightarrow v_c = \begin{cases} a \exp(\lambda_1 x) & z < 0 \\ \frac{\exp(\lambda_1 x)}{(a - p'(\lambda_1))} - (\frac{e^{(i\mathrm{Im}(\lambda_2)x)}}{(p'(\mathrm{Re}(\lambda_2) + i\mathrm{Im}(\lambda_2))} + \frac{e^{(-i\mathrm{Im}(\lambda_2)x)}}{p'(\mathrm{Re}(\lambda_2) - i\mathrm{Im}(\lambda_2))}) e^{(\mathrm{Re}(\lambda_2)x)} & 0 \leq z \leq z_1 \\ \frac{e^{\lambda_2(x - z_1)} - e^{(\lambda_2)x}}{p'(\lambda_2)} + \frac{e^{\lambda_3(x - z_1)} - e^{\lambda_3 x}}{p'(\lambda_3)} & z > z_1 \end{cases}, \tag{16}$$

$$\Rightarrow v_c = \begin{cases} a \exp(\lambda_1 x) & z < 0 \\ \frac{\exp(\lambda_1 x)}{(a - p'(\lambda_1))} - (\frac{\alpha - i\beta}{\alpha^2 + \beta^2} e^{(i\mathrm{Im}(\lambda_2)x)} + \frac{\alpha + i\beta}{\alpha^2 + \beta^2} e^{(-i\mathrm{Im}(\lambda_2)x)}) e^{(\mathrm{Re}(\lambda_2)x)} & 0 \leq z \leq z_1 \\ \frac{e^{\lambda_2(x - z_1)}}{p'(\lambda_2)} + \frac{e^{\lambda_3(x - z_1)}}{p'(\lambda_3)} - \left(\frac{e^{(\lambda_2)x}}{p'(\lambda_2)} + \frac{e^{\lambda_3 x}}{p'(\lambda_3)}\right) & z > z_1 \end{cases}, \tag{17}$$

$$\begin{aligned} \alpha &= 3(\mathrm{Re}(\lambda_2)^2 - (\mathrm{Im}(\lambda_2)^2) - 2\mathrm{Re}(\lambda_2) - 1, \\ \beta &= 6(\mathrm{Re}(\lambda_2))(\mathrm{Im}(\lambda_2)) - 2(\mathrm{Im}(\lambda_2)). \end{aligned}$$

Note that $\alpha, \beta \in \mathbb{R}$. Then applying Euler's formula

$$(a + bi)e^{-ci} + (a - bi)e^{ci} = 2(a\cos(c) + b\sin(c)) \qquad , a, b, c \in \mathbb{R}. \tag{18}$$

DOI: 10.1002/cnm

to write the formula in the region $0 \leq z \leq z_1$ as real expression

$$\Rightarrow v_c = \begin{cases} a\exp(\lambda_1 x) & z < 0 \\ \frac{\exp(\lambda_1 x)}{(a-p'(\lambda_1))} - 2(\frac{\alpha}{\alpha^2+\beta^2}\cos(\mathrm{Im}(\lambda_2)x) + \frac{\beta}{\alpha^2+\beta^2}\sin(\mathrm{Im}(\lambda_2)x))e^{(\mathrm{Re}(\lambda_2)x)} & 0 \leq z \leq z_1 \\ \frac{e^{\lambda_2(x-z_1)}}{p'(\lambda_2)} + \frac{e^{\lambda_3(x-z_1)}}{p'(\lambda_3)} - \left(\frac{e^{(\lambda_2)x}}{p'(\lambda_2)} + \frac{e^{\lambda_3 x}}{p'(\lambda_3)}\right) & z > z_1 \end{cases}, \tag{19}$$

$$\alpha = 3(\mathrm{Re}(\lambda_2)^2 - (\mathrm{Im}(\lambda_2)^2) - 2\mathrm{Re}(\lambda_2) - 1,$$
$$\beta = 6(\mathrm{Re}(\lambda_2))(\mathrm{Im}(\lambda_2)) - 2(\mathrm{Im}(\lambda_2)).$$

For the term in $z > z_1$ we apply twice the steps we used in the region $0 \leq z \leq z_1$, which results in the real expression

$$v_c = \begin{cases} a\exp(\lambda_1 x) & z < 0 \\ \frac{\exp(\lambda_1 x)}{(a-p'(\lambda_1))} - 2(\frac{\alpha}{\alpha^2+\beta^2}\cos(\mathrm{Im}(\lambda_2)x) + \frac{\beta}{\alpha^2+\beta^2}\sin(\mathrm{Im}(\lambda_2)x))e^{(\mathrm{Re}(\lambda_2)x)} & 0 \leq z \leq z_1 \\ 2(\frac{\alpha}{\alpha^2+\beta^2}\cos(\mathrm{Im}(\lambda_2)(x-z_1)) + \frac{\beta}{\alpha^2+\beta^2}\sin(\mathrm{Im}(\lambda_2)(x-z_1)))e^{(\mathrm{Re}(\lambda_2)(x-z_1))} - & z > z_1 \\ \quad -2(\frac{\alpha}{\alpha^2+\beta^2}\cos(\mathrm{Im}(\lambda_2)x) + \frac{\beta}{\alpha^2+\beta^2}\sin(\mathrm{Im}(\lambda_2)x))e^{(\mathrm{Re}(\lambda_2)x)} & \end{cases} \tag{20}$$

Rintzel and Keller showed that the above solution (15) holds true only if the parameters $a(b, c)$ satisfies the following relation with the parameter b and c. The relation assumes we know the eigenvalues λ_i $i = 1, 2, 3$ for given b, c, which then define the function

$$f(s) = 2 - s + \frac{p'(\lambda_1)}{p'(\lambda_2)}s^{(-\lambda_2/\lambda_1)} + \frac{p'(\lambda_1)}{p'(\lambda_3)}s^{(-\lambda_3/\lambda_1)}.$$

The root s_0 of the function f defines

$$a = \frac{1 - s_0}{p'(\lambda_1)}. \tag{21}$$

This relation can be satisfied for any b with at most two c_i, where $c_1 \leq c_2$. The slow pulse c_1 is an unstable solution, while c_2 is a stable solution [**?**, **?**]. To obtain the value of z_1 for the given set of parameter a, b, c the following equation needs to be solved

$$\exp(-\lambda_1 z_1 s_0) = 1 - ap'(\lambda_1). \tag{22}$$

3.2.1. Numerical simulation of the traveling wave For the verification of the dynamic solution we use the solution for $a = 0.2250646$ mV, $c = 1.2$ cm/ms , $b = 0.2$ and $z_1 = 6.63395$ cm. In the numerical problem the parameter $\sigma_i = 1967.5, \beta = 1$, $R = 0.1$ kΩ and $C_m = 1$ μF are used. We use the solution (15) to initialise our numerical solution at the time 0 ms on a line of length 160 cm, and origin at 85 cm. With these values the wave exits the domain at 50 ms. The final time is chosen such that the wave in the numerical simulation stabilises in shape and then propagates for about 20 ms. All simulation use a temporal time step of 0.001 ms.

The first experiment was performed for spatial resolution of 0.00625 mm, where the L^2-error was calculated at each time step and plotted against the time for all three solvers (Fig. 6). For the first iterations the error increases slower compared to the error increase after about 20 ms. Thereafter, a linear increase of the error is observed. This can be explained with the plot of the L^2-norm of the solution (Fig. 6), which is changing until 20 ms and thereafter can be considered as constant. The changes are due to the fact that the maximal amplitude of the wave is changing. The stable, slightly larger pulse, has after 20 ms a higher conduction velocity of about $c = 1.20132$ m/s, which is responsible for the linearly increasing error over time.

We conclude the verification with a convergence test in the L_2 error and the conduction velocity for the dynamic simulation. Therefore, the L_2 error and the conduction velocity after 40 ms have been evaluated for different step discretisations $h = \{1$ mm,

DOI: 10.1002/cnm

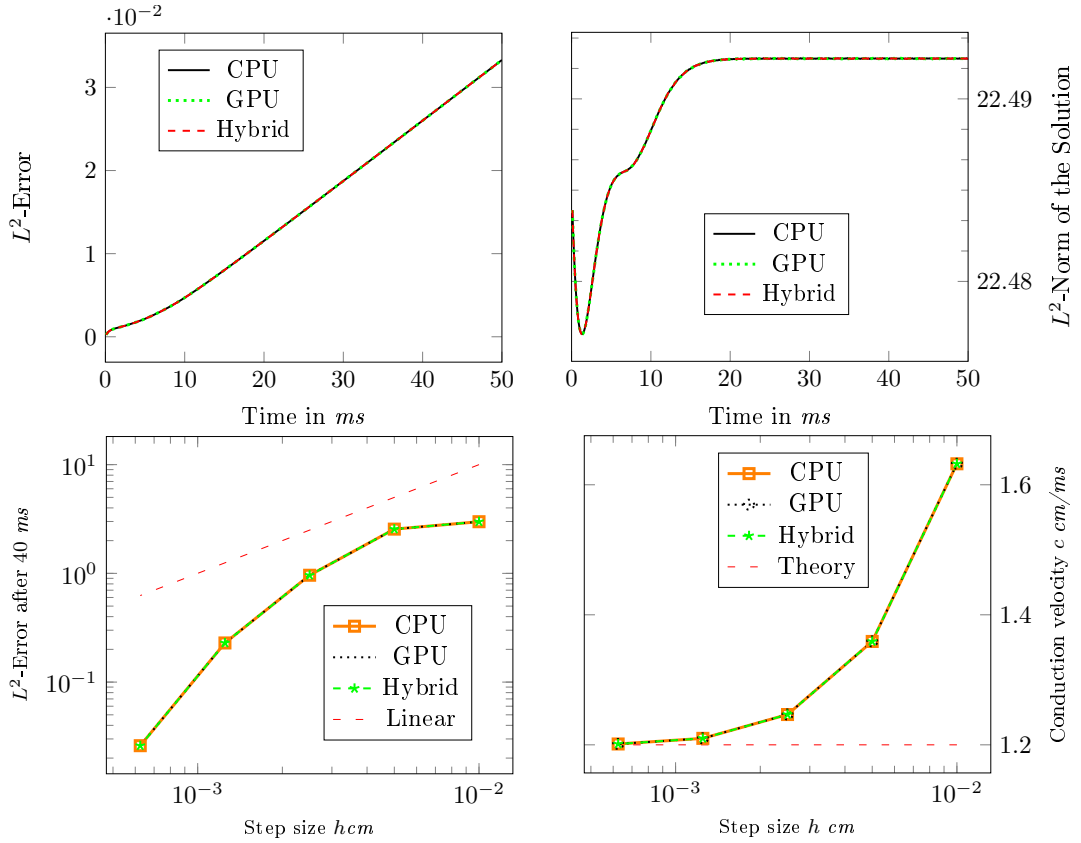

Figure 6. Simulation results with the traveling wave. On the upper panel the development of characteristics of the numerical solution over time is shown. Left hand the L^2 error and on the right L^2-Norm of the solution. The solution was obtained with the CPU (solid), hybrid (dashed) and the GPU (dotted), data shown for a spatial step size of 0.00625 mm. The lower panel shows at the left the L^2 error and at the right the conduction velocity after 40 *ms* against the step discretisation, obtained with the CPU (solid), hybrid (dashed) and the GPU (dotted)

0.5 mm, 0.25 mm, 0.125 mm, 0.0625 mm$\}$. The L_2 error converges superlinearly. More importantly, the conduction velocity approaches the theoretical value of 1.2 m/s at a step size of 0.0625 mm (Fig. 6). Again, the improved method of Sect. 2.1 remains consistent and subsequently exhibits proper convergence to the exact conduction velocity.

4. COMPUTATIONAL EFFICIENCY

Finally, the performance of the different implementations is evaluated to choose the most efficient one on problems of varying size and complexity. The number of DOFs are varied either by increasing the complexity of the Purkinje fibre network (spatial complexity), or by the switching to a more complex cell membrane model (model complexity).

The spatial complexity varies over four different Purkinje fibre networks, which are generated with the fractal rule presented in [**?**]. In order of increasing complexity, the first Purkinje network consists of the main Purkinje branches only, the second one has another level of branching giving a physiological covering of the LV, and the third network has another level of Purkinje branches added to increase the density of the end-junctions, resulting in a physiological network for the left ventricle. The fourth case is a dense Purkinje network for both the left and right ventricles.

DOI: 10.1002/cnm

All Purkinje networks have a spatial resolution of $0.1\,\mathrm{mm}$ and are generated without loops for compatibility with other solvers (see Fig. 7).

Two different cell membrane models were used to test the influence of model complexity. The first and simpler Di Francesco-Noble model [**?**], which has been used in previous works [**?**], has 15 state variables. The model has been obtained from the CellML database and used without modification to the initial states or constants. The second membrane model used here has been published by Stewart et al. [**?**], is based on modifications to the ten Tusscher-Panfilov model, and has 20 state variables. The model was obtained from CellML, and the initial conditions were changed (Appendix Tab. I). The change of the initial condition was necessary to avoid the early self-excitation that is present in Purkinje cells but should not manifest itself under physiological conditions. For both membrane models a cell length of 0.01 mm, cell radius of 0.00 5mm, and an intracellular conductivity of 40 $\Omega^{-1}\mathrm{cm}^{-1}$ were assumed, were the last two values are chosen to obtain a realistic conduction velocity between 3 m/s and 4 m/s. The gap junction resistance was chosen as 500 kΩ. For the simulation a temporal step size of 0.01 ms has been used and the simulation was run for 50 ms, after which all networks were fully depolarised. The meshes corresponding to the Purkinje networks can be downloaded as `VTK`[§] files, including the local activation times as supplemental material.

In the CPU implementation eight processes are run in parallel, while the hybrid and GPU implementations are run using one CPU process. The simulations were run in two different configurations on the GPU, the first in double precision, while the second was in single precision. For all simulations, we measure the time spend on setting up the problem, which includes reading the mesh and assembling the matrices and preconditioners. We report the time needed for solving the ionic membrane models and the diffusion equation in Fig. 7.

As expected the hybrid and GPU implementations are faster than the CPU implementation (Fig. 7), and a further speed-up is observed moving from double precision to single precision in the GPU implementation. The reason for the speed-up with the single precision in the GPU implementation is that the particular GPU used has roughly twice the number of floating point operations in single precision than it has in double precision. The reduction of computational time achieved with the hybrid implementation was only limited. One possible reason for this might be that the particular GPU used is able to handle the entire double precision problem without full occupation, and the second reason might be that the transmembrane potential needs to be converted from double precision to single precision, which is done in serial on the CPU.

The amount of time needed for the reaction part of the problem varies considerably between the CPU, hybrid, and GPU implementations. The CPU implementation is always the slowest, but the hybrid implementation performs more favourably on less complex membrane models, while the GPU implementation performs better with more complex membrane models. This is much more evident in the single precision versions. A possible reason for this can be found in the workflow of the hybrid and GPU implementations (Fig. 3), where a memory copy from the GPU to the CPU takes place in each time step of the hybrid implementation. In the GPU implementation this is unnecessary because values are used on the GPU only. This explains why the GPU implementation performs better with increasing complexity of the membrane model. Similar behaviour is observed for the diffusion step.

Solving the diffusion step with the GPU implementation is nearly always the slowest. We note that the hybrid implementation is faster than the CPU, as in the CPU implementation the transmembrane potential and the current need to be sent from all the OpenMPI nodes to the master node and the results communicated back. The linear system itself is solved in the same way in the CPU and the GPU/CPU hybrid cases, while in the GPU implementation the same algorithm is used, but the matrix operations are performed on the GPU. For small Purkinje systems, meaning very sparse and small matrices, the performance of the GPU implementation is behind the CPU and hybrid implementations. With increasing spatial complexity the GPU performance becomes better compared to the CPU performance, which likely is related to the size of the problem. Due to the

[§]The Visualization Toolkit (http://www.vtk.org)

DOI: 10.1002/cnm

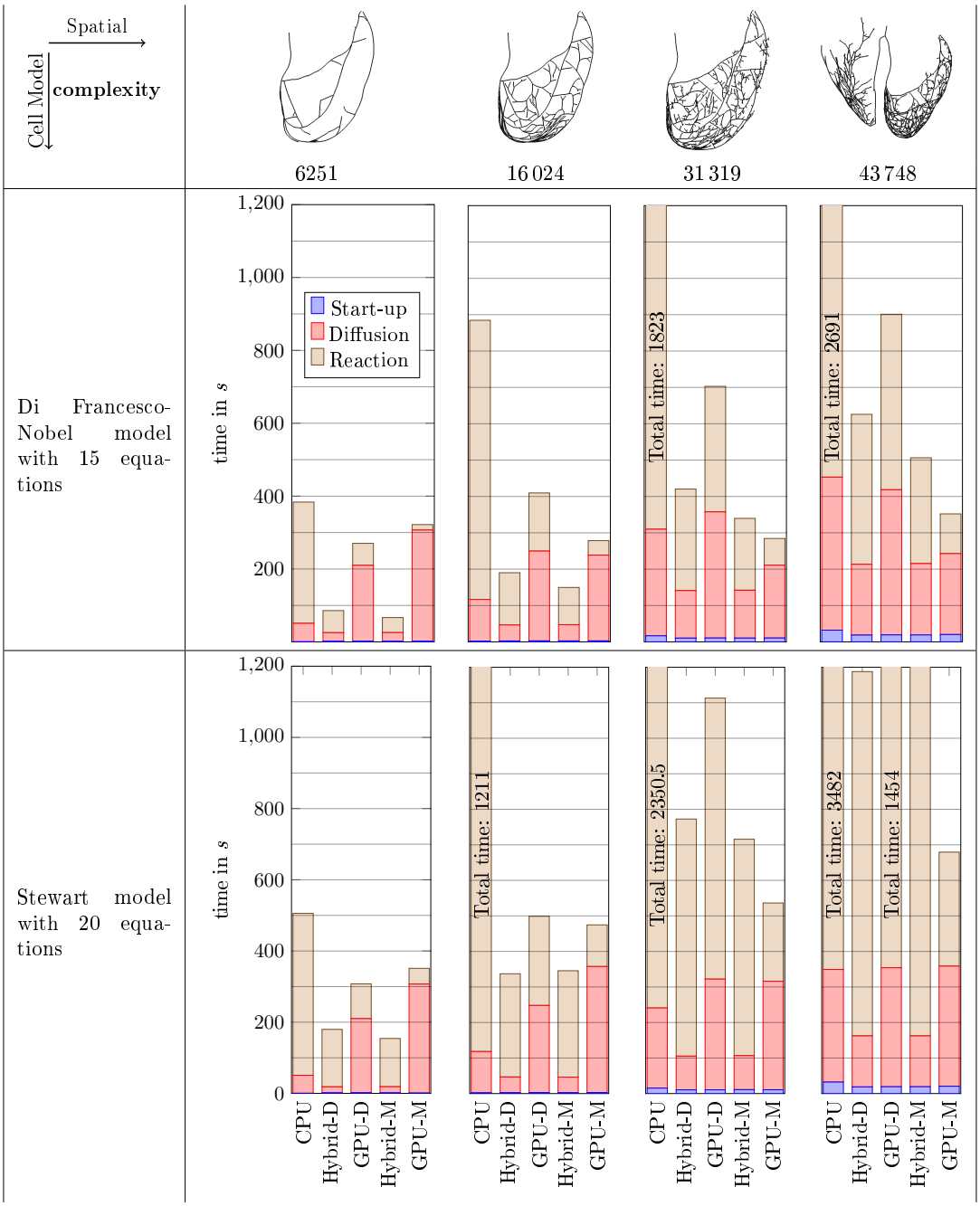

Figure 7. Performance test for two different cell models and four Purkinje networks.

overhead introduced by each CUDA operation, for very small problem sizes the benefits of GPU parallelism are lost.

5. CONCLUSION

We have presented an improved parallel algorithm for solving the monodomain cardiac electrophysiology equations on one-dimensional branching Purkinje fibre networks that is suitable

DOI: 10.1002/cnm

for simulating activation on realistic Purkinje fibre networks in human-size hearts. Three different implementations of the algorithm were tested: pure-CPU, pure-GPU, and hybrid. The main contributions of this work were (i) a modification to the previously published algorithm for the monodomain equations on Purkinje fibre networks with improved consistency at branching points,(ii) a change in the equivalent conductivity to be consistent with the cell length and mesh discretisation . (iii) formal verification of numerical solutions and their appropriate order of convergence, and (iiii) computational efficiency comparison of the three different implementations on Purkinje networks of varying spatial complexity coupled to different cell membrane models of increasing complexity.

The verification of the numerical solution showed no notable difference in convergence rates between the three different implementations.

The largest benefit of the parallel pure-GPU implementation was obtained either when a fully detailed biventricular (spatially complex) network was used, or when sufficiently complex membrane models were used, such as the Stewart et al. 2009 model considered in this study. For simpler LV-only models or when using simpler membrane models, such as the Di Francesco-Noble model considered in this study, the hybrid implementation may be more attractive. In either case the benefits of GPU-accelerated computation of action potentials in the fast conduction system have been demonstrated. While in this study we only considered the simplified monodomain equations, extensions could be made to incorporate similar algorithms for the bidomain equations to explore e.g. the effect of defibrillation on Purkinje fibres. Further work is also ongoing to extend the Purkinje cell model to account for the presence of ischemia in the heart. We hope to have convinced the cardiac modelling community that large-scale numerical simulation of action potentials in the Purkinje fibre system is both feasible and important.

ACKNOWLEDGEMENT

Simone Palamara has been funded by “Fondazione Cassa di Risparmio di Trento e Rovereto” (CARITRO) within the project “Numerical modelling of the electrical activity of the heart for the study of the ventricular dyssynchrony”. Christian Vergara has been partially supported by the Italian MIUR PRIN09 project no. 2009Y4RC3B_001.

DOI: 10.1002/cnm

APPENDIX

Table I. Initial conditions used for the Stewart et al. 2009 model

V	Transmembrane potential [mV]	-75.6095
K_i	Potassium dynamics [mMol]	136.757
Na_i	Sodium dynamics [mMol]	0.80211
Ca_i	Intracellular calcium [mMol]	1.47164e-4
y	y gate	0.00780153
X_{r1}	Rapid time dependent potassium current	0.382558
X_{r2}	Rapid time dependent potassium current	0.37373
X_s	Slow time dependent potassium current	3.85284e-2
m	m gate	1.24135e-2
h	h gate	0.361832
j	j gate	0.102063
Ca_{ss}	Calcium dynamics [mMol]	5.49319e-4
d	L type Ca current d	1.21585e-4
f	L type Ca current f	0.611603
f_2	L type Ca current $f2$	0.861484
f_{cass}	L-type Ca current	0.985735
s	Transient outward current s	0.925862
r	Transient outward current r	6.46602e-4
Ca_{SR}	Calcium in sarcoplasmic reticulum [mMol]	3.17519
R_{prime}	Calcium dynamics	0.851882

DOI: 10.1002/cnm

REFERENCES

1. A. Ansari, S.Y. Ho, and R.H. Anderson. Distribution of the Purkinje fibres in the sheep heart. *Anat. Rec.*, 254(1):92–97, 1999.
2. R. Artebrant, A. Tveito, and G.T Lines. A method for analyzing the stability of the resting state for a model of pacemaker cells surrounded by stable cells. *Math. Biosci. Eng.*, 7(3):505–526, 2010.
3. O. Berenfeld and J. Jalife. Purkinje-muscle reentry as a mechanism of polymorphic ventricular arrhythmias in a 3-dimensional model of the ventricles. *Circ. Res.*, 82(10):1063–1077, 1998.
4. F. Bogun, E. Good, S. Reich, D. Elmouchi, P. Igic, D. Tschopp, S. Dey, A. Wimmer, K. Jongnarangsin, H. Oral, A. Chugh, F. Pelosi, and F. Morady. Role of Purkinje fibers in post-infarction ventricular tachycardia. *J. Am. Coll. Cardiol.*, 48(12):2500–2507, 2006.
5. R.M. Bordas, K. Gillow, D. Gavaghan, B. Rodriguez, and D. Kay. A bidomain model of the ventricular specialized conduction system of the heart. *SIAM J. Appl. Math.*, 72(5):1618–1643, 2012.
6. P Colli Franzone, L.F. Pavarino, and S. Scacchi. *Mathematical cardiac electrophysiology*, volume 13 of *Modeling, Simulation & Applications*. Springer Cham Heidelberg New York Dordrecht London, 2014.
7. L.L. Cooper, K.E. Odening, M.-S. Hwang, L. Chaves, L. Schofield, C.A. Taylor, A.S. Gemignani, G.F. Mitchell, J.R. Forder, B.-R. Choi, and G. Koren. Electromechanical and structural alterations in the aging rabbit heart and aorta. *Am. J. Physiol. Heart Circ. Physiol.*, 302(8):H1625–H1635, 2012.
8. M. Deo, P.M. Boyle, A.M. Kim, and E.J. Vigmond. Arrhythmogenesis by single ectopic beats originating in the Purkinje system. *Am. J. Physiol. Heart Circ. Physiol.*, 299(4):H1002–H1011, 2010.
9. D. DiFrancesco and D. Noble. A model of cardiac electrical activity incorporating ionic pumps and concentration changes. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 307(1133):353–398, 1985.
10. J. Keener and J. Sneyd. *Mathematical Physiology I: Cellular Physiology*, volume 8(I) of *Interdisciplinary Applied Mathematics*. Springer New York, New York, NY, 2nd edition, 2009.
11. S. Krishnamoorthi, L.E. Perotti, N.P. Borgstrom, O.A. Ajijola, A. Frid, A.V. Ponnaluri, J.N. Weiss, Z. Qu, W.S. Klug, D.B. Ennis, et al. Simulation methods and validation criteria for modeling cardiac ventricular electrophysiology. *PlOS ONE*, 9(12):e114494, 2014.
12. M. Lange, S. Palamara, T. Lassila, C. Vergara, A. Quarteroni, and A.F. Frangi. Efficient numerical schemes for computing cardiac electrical activation over realistic purkinje networks: method and verification. *In: H. van Assen, P. Bovendeerd, T. Delhaas (Eds.) Proc. 8th Int. Conf. Functional Imag. Model. Heart (FIMH 2015), June 25–27, Maastricht*, 2015.
13. H.P McKean Jr. Nagumo's equation. *Adv. Math.*, 4(3):209–223, 1970.
14. A. Mena and J.F. Rodriguez. Using graphic processor units for the study of electric propagation in realistic heart models. In *IEEE Computing in Cardiology*, volume 39, pages 37–40, 2012.
15. M. Naumov. Parallel solution of sparse triangular linear systems in the preconditioned iterative methods on the GPU. Technical Report NVR-2011-001, NVIDIA, 2011.
16. S.A. Niederer, E. Kerfoot, A.P. Benson, M.O. Bernabeu, O. Bernus, C. Bradley, E.M. Cherry, R. Clayton, F.H. Fenton, A. Garny, E. Heidenreich, S. Land, M. Maleckar, P. Pathmanathan, G. Plank, J.F. Rodríguez, I. Roy, F.B. Sachse, G. Seemann, O. Skavhaug, and N.P. Smith. Verification of cardiac tissue electrophysiology simulators using an N-version benchmark. *Philos. Trans. A. Math. Phys. Eng. Sci.*, 369(1954):4331–51, 2011.
17. V. Nimmagadda, A. Akoglu, S. Hariri, and T. Moukabary. Cardiac simulation on multi-GPU platform. *J. Supercomput.*, 59(3):1360–1378, 2012.
18. A. Nogami. Purkinje-related arrhythmias part I: Monomorphic ventricular tachycardias. *Pacing Clin. Electrophysiol.*, 34(5):624–650, 2011.
19. S. Palamara, M. Lange, C. Vergara, T. Lassila, A.F. Frangi, and A. Quarteroni. A coupled 3D-1D numerical monodomain solver for cardiac electrical activation in the myocardium with detailed Purkinje network. Technical Report MOX-Report No. 19/2015, Politecnico di Milano, 2015.
20. S. Palamara, C. Vergara, D. Catanzariti, E. Faggiano, C. Pangrazzi, M. Centonze, F. Nobile, M. Maines, and A. Quarteroni. Computational generation of the Purkinje network driven by clinical measurements: the case of pathological propagations. *Med. Biol. Eng. Comput.*, 30(12):1558–1577, 2014.
21. P. Pathmanathan and R.A. Gray. Verification of computational models of cardiac electrophysiology. *Int. J. Numer. Methods Biomed. Engr.*, 30(5):525–544, 2014.
22. J. Rinzel and J.B. Keller. Traveling wave solutions of a nerve conduction equation. *Biophys. J.*, 13(12):1313–37, 1973.
23. R. Sebastian, V. Zimmerman, D. Romero, and A.F. Frangi. Construction of a computational anatomical model of the peripheral cardiac conduction system. *IEEE Trans. Biomed. Eng.*, 58(12):3479–82, 2011.
24. P. Stewart, O.V. Aslanidi, D. Noble, P.J. Noble, M.R. Boyett, and H. Zhang. Mathematical models of the electrical action potential of Purkinje fibre cells. *Philos. Trans. A. Math. Phys. Eng. Sci.*, 367(1896):2225–55, 2009.
25. K.H.W.J. Ten Tusscher and A.V. Panfilov. Modelling of the ventricular conduction system. *Prog. Biophys. Mol. Biol.*, 96(1):152–170, 2008.
26. A. Tveito and G.T. Lines. A condition for setting off ectopic waves in computational models of excitable cells. *Math. Biosci.*, 213(2):141–50, 2008.
27. C. Vergara, S. Palamara, D. Catanzariti, F. Nobile, E. Faggiano, C. Pangrazzi, M. Centonze, M. Maines, A. Quarteroni, and G. Vergara. Patient-specific generation of the Purkinje network driven by clinical measurements of a normal propagation. *Med. Biol. Eng. Comput.*, 52(10):813–826, 2014.
28. E.J. Vigmond and C. Clements. Construction of a computer model to investigate sawtooth effects in the Purkinje system. *IEEE Trans. Biomed. Eng.*, 54(3):389–99, 2007.

Recent publications:

MATHEMATICS INSTITUTE OF COMPUTATIONAL SCIENCE AND ENGINEERING
Section of Mathematics

Ecole Polytechnique Fédérale (EPFL)

CH-1015 Lausanne

32.2015 ASSYR ABDULLE, ONDREJ BUDÁČ:
A discontinuous Galerkin reduced basis numerical homogenization method for fluid flow in porous media

33.2015 ASSYR ABDULLE:
Numerical homogenization methods for parabolic monotone problems

34.2015 JÖRG LIESEN, ROBERT LUCE:
Fast recovery and approximation of hidden Cauchy structure

35.2015 ASSYR ABDULLE, MARTIN HUBER:
Numerical homogenization method for parabolic advection-diffusion multiscale problems with large compressible flows

36.2015 ASSYR ABDULLE, ORANE JECKER, ALEXANDER SHAPEEV:
An optimization based coupling method for multiscale problems

***

01.2016 ANDREA MANZONI, FEDERICO NEGRI, ALFIO QUARTERONI:
Dimensionality reduction of parameter-dependent problems through proper orthogonal decomposition

02.2016 SHENFENG ZHU, LUCA DEDÈ, ALFIO QUARTERONI:
Isogeometric analysis and proper orthogonal decomposition for the acoustic wave equation

03.2016 ROBERT LUCE, PETER HILDEBRANDT, UWE KUHLMANN, JÖRG LIESEN,:
Using separable non-negative matrix factorization techniques for the analysis of time-resolved Raman spectra

04.2015 ASSYR ABDULLE, TIMOTHÉE POUCHON:
Effective models for the multidimensional wave equation in heterogeneous media over long time and numerical homogenization

05.2016 ALFIO QUARTERONI, TONI LASSILA, SIMONE ROSSI, RICARDO RUIZ-BAIER:
Integrated heart – Coupling multiscale and multiphysics models for the simulation of the cardiac function

06.2016 M.G.C. NESTOLA, E. FAGGIANO, C. VERGARA, R.M. LANCELLOTTI, S. IPPOLITO, S. FILIPPI, A. QUARTERONI, R. SCROFANI :
Computational comparison of aortic root stresses in presence of stentless and stented aortic valve bio-prostheses

07.2016 M. LANGE, S. PALAMARA, T. LASSILA, C. VERGARA, A. QUARTERONI, A.F. FRANGI:
Improved hybrid/GPU algorithm for solving cardiac electrophysiology problems on Purkinje networks