

A matrix-free high-order solver for the numerical solution of cardiac electrophysiology

P. C. Africa^{a,*}, M. Salvador^a, P. Gervasio^b, L. Dede^a, A. Quarteroni^{a,c}

^a*MOX – Dipartimento di Matematica, Politecnico di Milano,
P.zza Leonardo da Vinci, 32, 20133 Milano, Italy*

^b*DICATAM, Università degli Studi di Brescia,
Via Branze, 38, 25123 Brescia, Italy*

^c*Mathematics Institute, École Polytechnique Fédérale de Lausanne,
Av. Piccard, CH-1015 Lausanne, Switzerland (Professor Emeritus)*

Abstract

We propose a matrix-free solver for the numerical solution of the cardiac electrophysiology model consisting of the monodomain nonlinear reaction-diffusion equation coupled with a system of ordinary differential equations for the ionic species. Our numerical approximation is based on the high-order Spectral Element Method (SEM) to achieve accurate numerical discretization while employing a much smaller number of Degrees of Freedom than first-order Finite Elements. We combine sum-factorization with vectorization, thus allowing for a very efficient use of high-order polynomials in a high performance computing framework. We validate the effectiveness of our matrix-free solver in a variety of applications and perform different electrophysiological simulations ranging from a simple slab of cardiac tissue to a realistic four-chamber heart geometry. We compare SEM to SEM with Numerical Integration (SEM-NI), showing that they provide comparable results in terms of accuracy and efficiency. In both cases, increasing the local polynomial degree p leads to better numerical results and smaller computational times than reducing the mesh size h . We also implement a matrix-free Geometric Multigrid preconditioner that entails better performance in terms of linear solver iterations than state-of-the-art matrix-based Algebraic Multigrid preconditioners. As a matter of fact, the matrix-free solver here proposed yields up to $50\times$ speed-up with respect to a conventional matrix-based solver.

Keywords: Cardiac electrophysiology, Matrix-free solver, Spectral Element Method, High Performance Computing, Geometric Multigrid

*Corresponding author.

Email address: pasqualeclaudio.africa@polimi.it (P. C. Africa)

1. Introduction

Mathematical and numerical modeling of cardiac electrophysiology provides meaningful tools to address clinical problems *in silico*, ranging from the cellular to the organ scale [1, 2, 3, 4, 5]. For this reason, several mathematical models and methods have been designed to perform electrophysiological simulations [6, 7]. Among these, we consider the monodomain equation coupled with suitable ionic models, which describes the space–time evolution of the transmembrane potential and how chemical species move across ionic channels [8].

This set of combined partial and ordinary differential equations describes solutions that resemble those of a wavefront propagation problem, *i.e.* manifesting very steep gradients. Despite being extensively used [9, 10, 11, 12], the Finite Element Method (FEM) with first order polynomials does not seem to be the most suitable to properly capture the physical processes underlying cardiac electrophysiology [13]. Indeed, in such cases, a very fine mesh resolution is required to obtain fully convergent numerical results [14], which calls for an overwhelming computational burden.

High–order numerical methods come into play to tackle this specific issue: Spectral Element Method (SEM) [15, 16, 17], high–order Discontinuous Galerkin (DG) [18, 19], Finite Volume Method (FVM) [20], or Isogeometric Analysis (IGA) [21] account for small numerical dispersion and dissipation errors while allowing for converging numerical solutions with less Degrees of Freedom (DOFs) [22, 23, 24, 25]. However, the use of high–order polynomials in *matrix–based* solvers for complex scenarios has been hampered by several numerical challenges, which are mostly related to the stiffness of the discretized monodomain problem [26].

In this context, we develop and implement a high–order *matrix–free* numerical solver that can be readily employed for CPU–based, massively parallel, large–scale numerical simulations. Since there is no need to assemble any matrix, all the floating point operations are associated with matrix–vector products that represent the most demanding computational kernels at each iteration of iterative solvers. Thanks to vectorization [27], which enables algebraic operations on multiple mesh cells at the same time, and sum–factorization [28, 29], the higher the polynomial degree, the higher the computational advantages provided by the matrix–free solver [23, 30]. Moreover, the small memory occupation required by the matrix–free implementation allows for its exploitation in GPU–based cardiac solvers [31, 32, 33].

In this manner, we obtain very accurate and efficient numerical simulations for cardiac electrophysiology, even if the linear solver remains unpreconditioned. Additionally, we implement a matrix–free Geometric Multigrid (GMG) preconditioner that is optimal for the values of h (mesh size) and p (polynomial degree) considered in this paper when continuous model properties (*i.e.* a single ionic model and a continuous set of conductivity coefficients) are employed throughout the computational domain.

We present different benchmark problems of increasing complexity for cardiac electrophysiology, ranging from the *Niederer benchmark* on a slab of cardiac

tissue [34] to a whole-heart numerical simulation. We focus on two high-order discretization methods, namely, we compare SEM to SEM with Numerical Integration (SEM-NI), following the notations introduced in [17]. These two methods differ in the use of quadrature formulas: Legendre-Gauss formulae for SEM, Legendre-Gauss-Lobatto for SEM-NI. Numerical results of Section 5 show that the two methods feature a similar behaviour in terms of both accuracy and computational costs. In both cases, choosing a higher polynomial degree p leads to a fairly more beneficial ratio between accuracy and computational costs than reducing the mesh size h . For instance, working with two discretizations with the same number of DOFs on the Niederer benchmark, the solution computed with \mathbb{Q}_4 (local polynomials of degree 4 with respect to each spatial coordinate) and average mesh size $h_{\text{avg}} = 0.48$ mm is more accurate than the one obtained with \mathbb{Q}_1 and average mesh size $h_{\text{avg}} = 0.12$ mm. Moreover, the former one has been computed at a computational cost that is about 40% of the latter one.

We also evaluate the performance of our matrix-free solver: a $50\times$ speed-up is achieved with respect to the matrix-based solver. Furthermore, while the assembling and solving phases of the monodomain problem take more than 70% of the total computational time with the matrix-based implementation, this value drops to approximately 20% with the matrix-free solver.

The mathematical models and the numerical methods contained in this paper have been implemented in `lifex` (<https://lifex.gitlab.io/>), a high-performance C++ library developed within the iHEART project and based on the `deal.II` (<https://www.dealii.org>) Finite Element core [35].

The outline of the paper is as follows. We describe the monodomain model in Section 2. We address its space and time discretizations in Section 3. We propose the matrix-free solver for cardiac electrophysiology and the matrix-free GMG preconditioner in Section 4. Finally, the numerical results in Section 5 prove the high efficiency of our high-order SEM matrix-free solver against the low-order FEM matrix-based one.

2. Mathematical model

For the mathematical modeling of cardiac electrophysiology, we consider the monodomain equation coupled with suitable ionic models ([6, 8])

$$\left\{ \begin{array}{ll} \chi \left(C_m \frac{\partial u}{\partial t} + \mathcal{I}_{\text{ion}}(u, \mathbf{w}, \mathbf{z}) \right) - \nabla \cdot (\mathbf{D}_M \nabla u) = \chi \mathcal{I}_{\text{app}}(\mathbf{x}, t), & \text{in } \Omega \times (0, T], \\ (\mathbf{D}_M \nabla u) \cdot \mathbf{n} = 0, & \text{on } \partial\Omega \times (0, T], \\ \frac{d\mathbf{w}}{dt} = \mathbf{H}(u, \mathbf{w}, \mathbf{z}), & \text{in } \Omega \times (0, T], \\ \frac{d\mathbf{z}}{dt} = \mathbf{G}(u, \mathbf{w}, \mathbf{z}), & \text{in } \Omega \times (0, T], \\ u(\mathbf{x}, 0) = u_0(\mathbf{x}), \mathbf{w}(\mathbf{x}, 0) = \mathbf{w}_0(\mathbf{x}), \mathbf{z}(\mathbf{x}, 0) = \mathbf{z}_0(\mathbf{x}), & \text{in } \Omega. \end{array} \right. \quad (1)$$

The unknowns are: the transmembrane potential u , the vector $\mathbf{w} = (w_1, \dots, w_M)$ of the probability density functions of M gating variables, which represent the

fraction of open channels across the membrane of a single cardiomyocyte, and the vector $\mathbf{z} = (z_1, \dots, z_P)$ of the concentrations of P ionic species. For the sake of simplifying the notation, in the following the membrane capacitance per unit area C_m and the membrane surface-to-volume ratio χ are set equal to 1.

The mathematical expressions of the functions $\mathbf{H}(u, \mathbf{w}, \mathbf{z})$ and $\mathbf{G}(u, \mathbf{w}, \mathbf{z})$, which describe the dynamics of gating variables and ionic concentrations respectively, and the ionic current $\mathcal{I}_{\text{ion}}(u, \mathbf{w}, \mathbf{z})$ strictly depend on the choice of the ionic model. Here, the TTP06 [36] ionic model is adopted for the slab and ventricular geometries, while the CRN [37] ionic model is employed for the atria. The action potential is triggered by an external applied current $\mathcal{I}_{\text{app}}(\mathbf{x}, t)$.

The diffusion tensor \mathbf{D}_M is expressed as follows

$$\mathbf{D}_M = \sigma_1 \mathbf{f}_0 \otimes \mathbf{f}_0 + \sigma_t \mathbf{s}_0 \otimes \mathbf{s}_0 + \sigma_n \mathbf{n}_0 \otimes \mathbf{n}_0, \quad (2)$$

where the vector fields \mathbf{f}_0 , \mathbf{s}_0 and \mathbf{n}_0 express the fiber, the sheetlet and the sheet-normal (cross-fiber) directions, respectively. We also define longitudinal, transversal and normal conductivities as $\sigma_1, \sigma_t, \sigma_n \in \mathbb{R}^+$, respectively [38].

In this paper, the computational domain $\Omega \subset \mathbb{R}^3$ is represented either by a slab of cardiac tissue or by the Zygote geometry [39]. Homogeneous Neumann boundary conditions are prescribed on the whole boundary $\partial\Omega$ to impose the condition of electrically isolated domain, \mathbf{n} being the outward unit normal vector to the boundary.

3. Space and time discretizations

In order to discretize in space the system (1), we adopt SEM [15, 16, 17, 40, 41], a high-order method that can be recast in the framework of the Galerkin method [13].

We consider a family of hexahedral conforming meshes, satisfying standard assumption of regularity and quasi-uniformity [13], and let $h > 0$ denote the mesh size.

At each time, we look for the discrete solution belonging to the space of globally continuous functions that are the tensorial product of univariate piecewise (on each mesh element) polynomial functions of local degree $p \geq 1$ with respect to each coordinate. The local finite element space is referred to as \mathbb{Q}_p , while we denote by $V_{h,p}$ the global finite dimensional space.

When using SEM, the univariate basis functions are of Lagrangian (*i.e.*, nodal) type and their support nodes x_i are the Legendre–Gauss–Lobatto quadrature nodes (see, *e.g.*, [42, Ch. 2]), suitably mapped from the reference interval $[-1, 1]$ to the local 1D elements.

One of the main features of SEM is that, when the data are smooth enough, the induced approximation error decays more than algebraically fast with respect to the local polynomial degree. Indeed, it is said that SEM features exponential or spectral convergence. At the same time, the convergence with respect to the mesh size h behaves as in FEM. More precisely, if $u \in H^s(\Omega)$,

with $s > \frac{3}{2}$, denotes the exact solution of a linear second-order elliptic problem in a Lipschitz domain Ω and $u_{\text{SEM}} \in V_{hp}$ is its SEM approximation, the following error estimate holds

$$\|u - u_{\text{SEM}}\|_{H^1(\Omega)} \leq Ch^{\min(p+1,s)-1} p^{1-s} \|u\|_{H^s(\Omega)}. \quad (3)$$

SEM can be considered as a special case of hp -FEM ([41, 43, 44]) with nodal basis functions and conforming hexahedral meshes.

Typically, when using SEM, the integrals appearing in the Galerkin formulation of the differential problem (1) are computed by the composite Legendre–Gauss (LG) quadrature formulas (see [17, 42]). In principle, one can choose LG formulas of the desired order of exactness to guarantee a highly accurate computation of all the integrals appearing in (1). However, a typical choice is to use LG formulas with $(p + 1)$ quadrature nodes, which guarantees that the entries of both the mass matrix and the stiffness matrix with constant coefficients are computed exactly while keeping the computational costs not too large [30, 45].

A considerable improvement in reducing the computational times of evaluating the integrals consists of using Legendre–Gauss–Lobatto (LGL) quadrature formulas (instead of LG ones), again with $(p + 1)$ nodes that now coincide with the support nodes of the Lagrangian basis functions.

This results into the so-called SEM–NI method (NI standing for numerical integration). Since the Lagrangian basis functions are mutually orthogonal with respect to the discrete L^2 -inner product induced by the LGL formulas, the mass matrix of the SEM–NI method is diagonal, although not integrated exactly; this is a great strength of SEM–NI in solving time-dependent differential problems through explicit methods when the mass matrix is assembled. On the other hand, as the degree of exactness of LGL quadrature formulas using $(p + 1)$ nodes along each direction is $2p - 1$, the integrals associated with the nonlinear terms of the differential problem may introduce quadrature errors and aliasing effects that are as significant as the nonlinearities.

We remark that \mathbb{Q}_1 -SEM is equivalent to \mathbb{Q}_1 -FEM, while \mathbb{Q}_1 -SEM–NI is in fact \mathbb{Q}_1 -FEM in which the integrals are approximated by the trapezoidal quadrature rule [13].

We choose the same local polynomial degree p (and then the same finite dimensional space) for approximating the transmembrane potential u , the gating variables w_i (for $i = 1, \dots, M$) and the ionic concentrations z_i (for $i = 1, \dots, P$) at each time $t \in (0, T]$.

All the time derivatives appearing in Eq. (1) have been approximated by the 2nd-order Backward Differentiation Formula (BDF2) over a discrete set of time steps $t^n = n\Delta t$, $n = 0, \dots, N$, being Δt the time step size.

Regardless of the quadrature formula (LG or LGL), the algebraic counterpart of the monodomain problem (1) reads: given \mathbf{w}^0 , \mathbf{z}^0 and \mathbf{u}^0 , and suitable initializations for \mathbf{w}^1 , \mathbf{z}^1 and \mathbf{u}^1 , then, for any $n \geq 1$, find \mathbf{w}^{n+1} , \mathbf{z}^{n+1} and \mathbf{u}^{n+1}

by solving the following partitioned scheme:

$$\begin{cases} \frac{3\mathbf{w}^{n+1} - 4\mathbf{w}^n + \mathbf{w}^{n-1}}{2\Delta t} = \mathbf{H}(\mathbf{u}^*, \mathbf{w}^{n+1}, \mathbf{z}^{n+1}), \\ \frac{3\mathbf{z}^{n+1} - 4\mathbf{z}^n + \mathbf{z}^{n-1}}{2\Delta t} = \mathbf{G}(\mathbf{u}^*, \mathbf{w}^{n+1}, \mathbf{z}^{n+1}), \end{cases} \quad (4)$$

$$\mathbf{M} \frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t} + \mathbf{K}\mathbf{u}^{n+1} = \mathbf{s}^{n+1} + \mathbf{f}^{n+1}.$$

The arrays \mathbf{u}^{n+1} and \mathbf{w}^{n+1} and \mathbf{z}^{n+1} contain the SEM or SEM–NI DOFs of the transmembrane potential, gating variables and ionic concentrations, respectively, \mathbf{M} and \mathbf{K} are the SEM or SEM–NI mass and stiffness matrices, respectively, and $\mathbf{f}_i^{n+1} = \mathcal{I}_{\text{app}}(\mathbf{x}_i, t^{n+1})$. The entries of \mathbf{s}^{n+1} are computed with Ionic Current Interpolation (ICI) [46], *i.e.*,

$$s_i^{n+1} = - \int_{\Omega} \left(\sum_j \mathcal{I}_{\text{ion}}(\mathbf{u}_j^*, \mathbf{w}_j^{n+1}, \mathbf{z}_j^{n+1}) \varphi_j \right) \varphi_i, \quad (5)$$

with φ_i the i^{th} Lagrange basis function of the finite dimensional space $V_{h,p}$. We remark that, when SEM–NI with LGL quadrature formulas are employed, ICI coincides with Lumped–ICI [47], as the lumping of the SEM mass matrix coincides with the SEM–NI mass matrix.

If we set $\mathbf{u}^* = \mathbf{u}^{n+1}$, then we recover the fully implicit BDF2 scheme. Nevertheless, we highlight that the function \mathcal{I}_{ion} is typically strongly nonlinear. To overcome the drawbacks of this nonlinearity, we adopt the extrapolation formula $\mathbf{u}^* = 2\mathbf{u}^n - \mathbf{u}^{n-1}$ of \mathbf{u}^{n+1} , that is second–order accurate with respect to Δt . The resulting semi–implicit scheme is 2nd–order accurate in time when $\Delta t \rightarrow 0$ (see, *e.g.*, [48]).

The ordinary differential equations (4)_{1,2} are associated with the ionic model and provide both the gating variables and the ionic species, while equation (4)₃ is the discretization of the monodomain equation and its solution at the generic time step $n \geq 1$ is obtained by solving the linear system

$$\mathbf{A}\mathbf{u}^{n+1} = \mathbf{b}^{n+1}, \quad (6)$$

where

$$\mathbf{A} = \frac{3}{2\Delta t} \mathbf{M} + \mathbf{K}, \quad \mathbf{b}^{n+1} = \frac{1}{\Delta t} \mathbf{M}(2\mathbf{u}^n - \mathbf{u}^{n-1}) + \mathbf{s}^{n+1} + \mathbf{f}^{n+1}. \quad (7)$$

Solving the system (6) represents the most computationally demanding part of (4). We refer to Section 5, in particular to Table 8, for further details about this specific aspect.

4. Matrix–free and matrix–based solvers

As in FEM, the matrix \mathbf{A} based on either SEM or SEM–NI has a very sparse structure, thus iterative methods are the natural candidates to solve the linear

system (6). Since A is symmetric and positive definite, we have adopted the Conjugate Gradient (CG) method or its preconditioned version (PCG).

Excluding the preconditioner step, the most expensive part of one CG-iteration is the evaluation of a matrix-vector product $A\mathbf{v}$, where \mathbf{v} is a given vector.

Typically, in a conventional matrix-based solver, the matrix A is assembled and stored in sparse format, then referenced whenever the matrix-vector product has to be evaluated, *i.e.* during each iteration of the CG. The matrix-based solver aims at minimizing the number of floating points operations required for such evaluation and is a winning strategy in \mathbb{Q}_1 -FEM discretization for which the band of the matrix A is small.

When SEM (or SEM-NI) of local degree p are employed in the discretization process, each cell counts $(p + 1)^3$ DOFs. It follows that the typical bandwidth of SEM (or SEM-NI) stiffness matrices is about $C(p + 1)^3$ (where C is the maximum number of cells sharing one node of the mesh) and it exceeds widely that of \mathbb{Q}_1 -FEM stiffness matrices. The large bandwidth of the SEM matrix A can worsen the computational times of accessing the matrix entries, thus deteriorating the efficiency of the iterative solver.

Moreover, in modern processors, access to the main memory has become the bottleneck in many solvers for partial differential equations: a matrix-vector product based on matrices requires far more time waiting for data to arrive from memory than on actually doing the floating point operations. Thus, it is proved to be more efficient to recompute matrix entries – or rather, the action of the differential operator represented by these entries on a known vector, cell by cell – rather than looking up matrix entries in the memory, even if the former approach requires a significant number of additional floating point operations [30].

This approach is referred to as *matrix-free*. In practice, shape functions values and gradients are pre-computed for each basis function on the reference cell, for each quadrature node. Then, the Jacobian of the transformation from the real to the reference cell is cached, thus improving the computational cost of the evaluation.

A *matrix-free* solver can also benefit from *vectorization*, that brings an additional speedup. In FEM solvers (and, similarly, in SEM ones), the cell-wise computations are typically exactly the same for all cells, and hence a Single-Instruction, Multiple-Data (SIMD) stream can be used to process several values at once (see Figure 1). Vectorization is a SIMD concept, that is, one CPU instruction is used to process multiple cells at once. Modern CPUs support SIMD instruction sets to different extents, *i.e.* one single CPU instruction can simultaneously process from two doubles (four floats) up to eight doubles (or sixteen floats), depending on the underlying architecture [49]. Vectorization can be easily applied to a parallel framework [50], which in our case results in the scheme shown in Figure 2.

Vectorization is typically not used explicitly in *matrix-based* Finite Element codes as it is beneficial only in arithmetic intensive operations, whereas additional computational power becomes useless when the workload bottleneck is

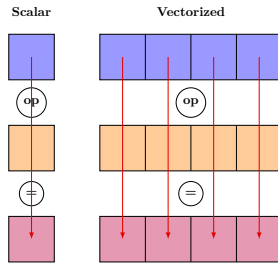


Figure 1: Comparison between scalar and vectorized operations.

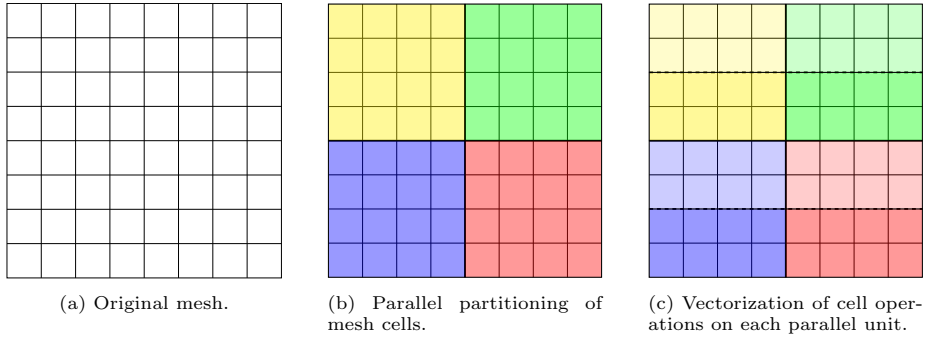


Figure 2: Vectorization in a parallel framework: when the original mesh is partitioned among multiple computational units, vectorization is applied at the level of each process. Here, different colors refer to different parallel units and light/dark variations represent different vectorized batches (in this example, vectorization acts on 8 cells).

the memory bandwidth.

When vectorization is available, it can be more convenient to repeat the arithmetic operations to compute the action of the matrix on a known vector every time that it is needed, instead of accessing the matrix entries to compute a matrix–vector product. Moreover, thanks to the fact that the multivariate SEM Lagrange basis is of tensorial type, in order to reduce the computational complexity of one evaluation of the product $\mathbf{A}\mathbf{v}$, *sum-factorization* can be exploited [28, 29, 51]; in this way, the matrix–vector product in the generic three dimensional cell requires only $9(p+1)$ floating point operations instead of $(p+1)^3$ per degree of freedom, resulting in a complexity equal to $\mathcal{O}(9(p+1)^4)$ instead of $\mathcal{O}((p+1)^6)$, and this plays in favor of repeating the computation rather than of accessing the memory (see [52, Sect. 2.3.1] and [42, Sect. 4.5.1]).

On these bases, a high–order matrix–free solver is more efficient both in terms of memory occupation (no system matrix is assembled and stored globally) and computational time [30], as we will also show in Sect. 5.

To precondition the CG method we have chosen Multigrid preconditioners. For the matrix–based solver, the Algebraic Multigrid (AMG) preconditioner [53, 54] turns out to be a very efficient choice. Nevertheless, its implementation requires the explicit knowledge of the entries of the matrix \mathbf{A} . For this reason, this preconditioner cannot be used in a matrix–free context.

To overcome this drawback, in the latter case we have adopted a Geometric Multigrid (GMG) preconditioner, more precisely the *high-order h-multigrid* preconditioner [55], which uses p –degree interpolation and restriction among geometrically coarsened meshes. GMG methods are among the most efficient solvers for linear systems arising from the discretization of elliptic partial differential equations, offering an optimal complexity $\mathcal{O}(n)$ in the number of unknowns n , and they are often used as very efficient preconditioners (see [32, 53, 56, 57] and the literature cited therein).

GMG turns out to be very efficient in a matrix–free context because all its computational kernels, including the Chebyshev smoother and the transfer between different grid levels, are based on matrix–vector products involving suitable collections of mesh cells [58]. In our implementation, the GMG preconditioner exploits the hierarchical meshes that are built by the recursive subdivision of each cell into 8 subcells, starting from a coarse mesh T_0 of size h_0 , as shown in Figure 3.

5. Numerical results

We present several numerical simulations of cardiac electrophysiology. First, we consider a benchmark problem on a slab of cardiac tissue [34], in order to compare SEM against SEM–NI and matrix–free against matrix–based in terms of computational efficiency and mathematical accuracy. Then, we employ the Zygote left ventricle geometry and we analyze the sole impact of increasing p , *i.e.* the local polynomial degree, on the numerical solution. Finally, for the sake of completeness, we show the capability of our matrix–free solver by presenting

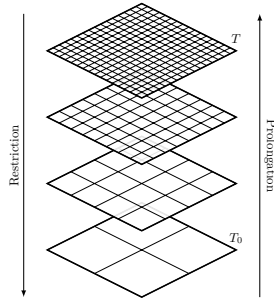


Figure 3: Schematization of multigrid methods. Starting from the real mesh T , the action of the *restriction* and *prolongation* operators is shown, down to the coarse mesh T_0 .

Variable	Value	Unit	Variable	Value	Unit
Conductivity tensor			Applied current		
σ_l	$0.7643 \cdot 10^{-4}$	$\text{m}^2 \text{s}^{-1}$	$\tilde{\mathcal{I}}_{\text{app}}^{\text{max}}$	15	V s^{-1}
σ_t	$0.3494 \cdot 10^{-4}$	$\text{m}^2 \text{s}^{-1}$	t_{app}	$3 \cdot 10^{-3}$	s
σ_n	$0.1125 \cdot 10^{-4}$	$\text{m}^2 \text{s}^{-1}$			

Table 1: Parameters of the electrophysiological model. For the CRN and TTP06 ionic models, we adopt the parameters reported in the original papers [36, 37] for epicardial cells.

a detailed Zygote four-chamber heart electrophysiological simulation in sinus rhythm.

For the time discretization, we use the BDF2 scheme with a time step $\Delta t = 0.1$ ms. The final time T differs with the specific test case. We employ $T = 0.2$ s in the Niederer benchmark [34], while $T = 0.6$ s and $T = 0.8$ s are considered for the left ventricle and whole-heart geometries, respectively.

Regarding the linear algebra back-end, we use the PCG solver with a stopping criterion based on the absolute residual with tolerance 10^{-15} . In the two test cases involving the slab and left ventricle, we employ the GMG (AMG, resp.) preconditioner for the matrix-free (matrix-based, resp.) solver. On the other hand, no preconditioner is introduced in the four-chamber heart numerical simulation, due to the presence of different ionic models in the computational domain, namely the CRN model ([37]) for atria and the TTP06 one ([36]) for ventricles.

Fiber generation is performed by means of the Laplace-Dirichlet rule-based methods proposed in [38, 59], while in Table 1 we report the parameters of the monodomain equation.

The external current $\mathcal{I}_{\text{app}}(\mathbf{x}, t) = \tilde{\mathcal{I}}_{\text{app}}^{\text{max}}$ is applied for $t \in (0, t_{\text{app}}]$ in a cuboid for the Niederer benchmark (as described in [34]), otherwise in different spheres for the ventricle and whole-heart test cases (we can deduce them from the numerical results shown in Figures 4, 10 and 12).

All the numerical simulations were performed by using one cluster node endowed with 56 cores (two Intel Xeon Gold 6238R, 2.20 GHz), which is available

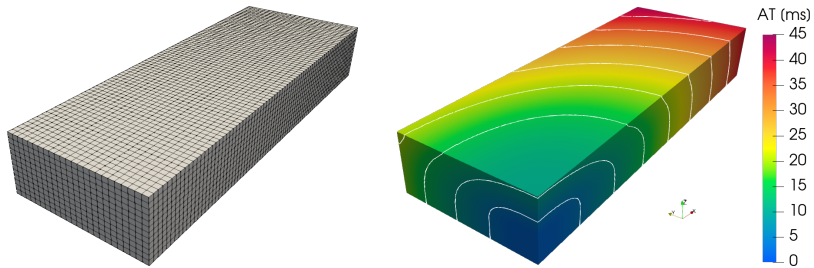


Figure 4: *Niederer benchmark*. The geometry and an example of mesh (left) and the associated simulated activation map (right). The blue color represents the region where the cubic stimulus is initially applied; the red one is associated with the corresponding diagonally opposite vertex, which is activated as last.

at MOX, Dipartimento di Matematica, Politecnico di Milano.

5.1. Slab of cardiac tissue

The computational domain with an example of mesh (left) and the associated numerical simulation (right) for the Niederer benchmark [34] is depicted in Figure 4. An external stimulus of cubic shape is applied at one vertex, the electric signal propagates through the slab, and the diagonally opposite vertex is activated as the last point. The domain is discretized by means of a structured, uniform hexahedral mesh.

We present a systematic comparison between SEM and SEM–NI for several values of both the mesh size h and the local polynomial degree p , in order to understand which is the best formulation in terms of accuracy and computational cost. Moreover, we compare the efficiency of the matrix-free and matrix-based solvers for SEM.

In Figures 5 and 6 we show the action potential and the calcium concentration computed with SEM and SEM–NI, respectively, over time. More precisely, the minimum, average, maximum and point values are plotted, where the max, min, and mean functions are evaluated on the set of nodes of the mesh. We notice that the convergence is faster for increasing p rather than for vanishing h .

At each node \mathbf{x}_i of the mesh, we also compute the activation time τ as the time instant when the approximation of the transmembrane potential u presents maximum derivative, *i.e.*

$$\tau(\mathbf{x}_i) = \arg \max_t \left| \frac{\partial u}{\partial t}(\mathbf{x}_i, t) \right|. \quad (8)$$

In the formula above t spans over the discrete set of time steps and the time derivative is approximated via the same scheme used for the time discretization of problem (1).

In Figure 7 we show the activation times along the slab diagonal, for different choices of the local space (from \mathbb{Q}_1 to \mathbb{Q}_4) and mesh refinements. As the error

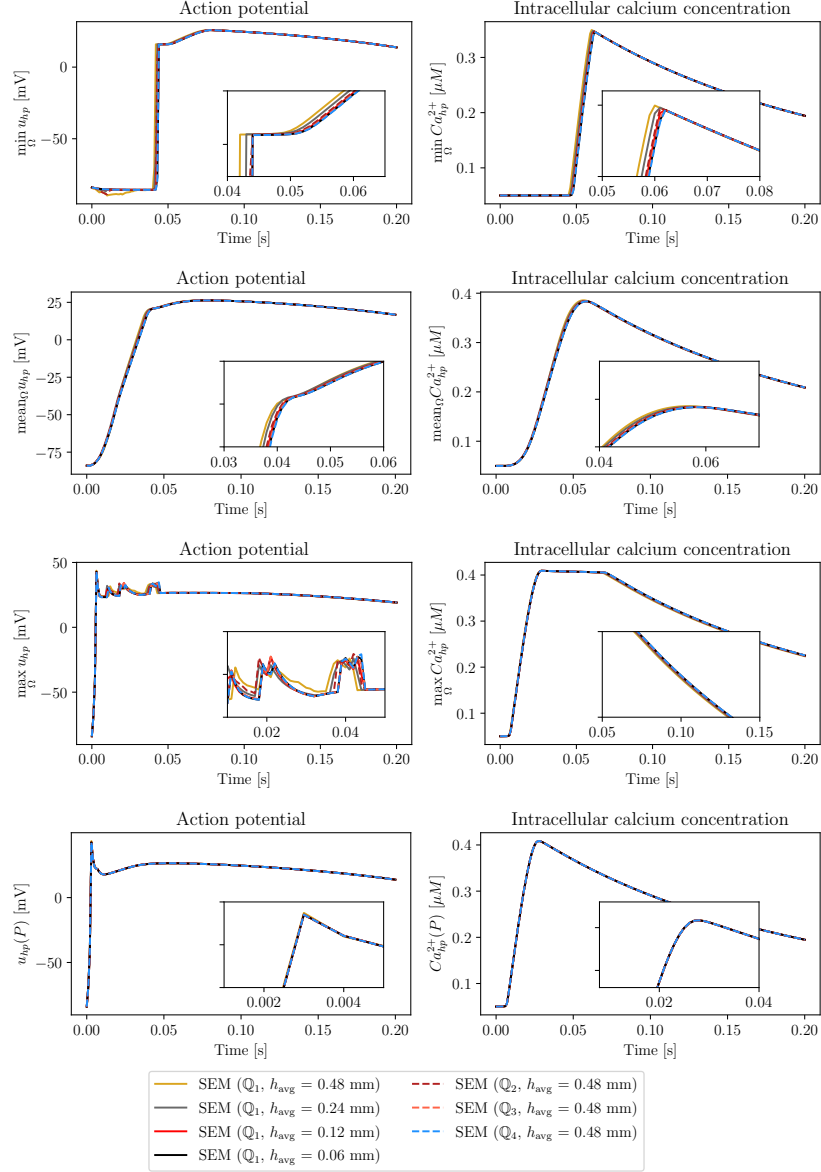


Figure 5: *Niederer benchmark*. Minimum (top), average (second), maximum (third) and point values (bottom) of the action potential u and intracellular calcium concentration Ca^{2+} over time for a slab of cardiac tissue. P is a random point within the computational domain away from the initial stimulus. We consider different hp combinations: SEM \mathbb{Q}_1 to \mathbb{Q}_4 and $h_{avg} = 0.48$ mm to $h_{avg} = 0.06$ mm (h_{avg} is the average mesh size).

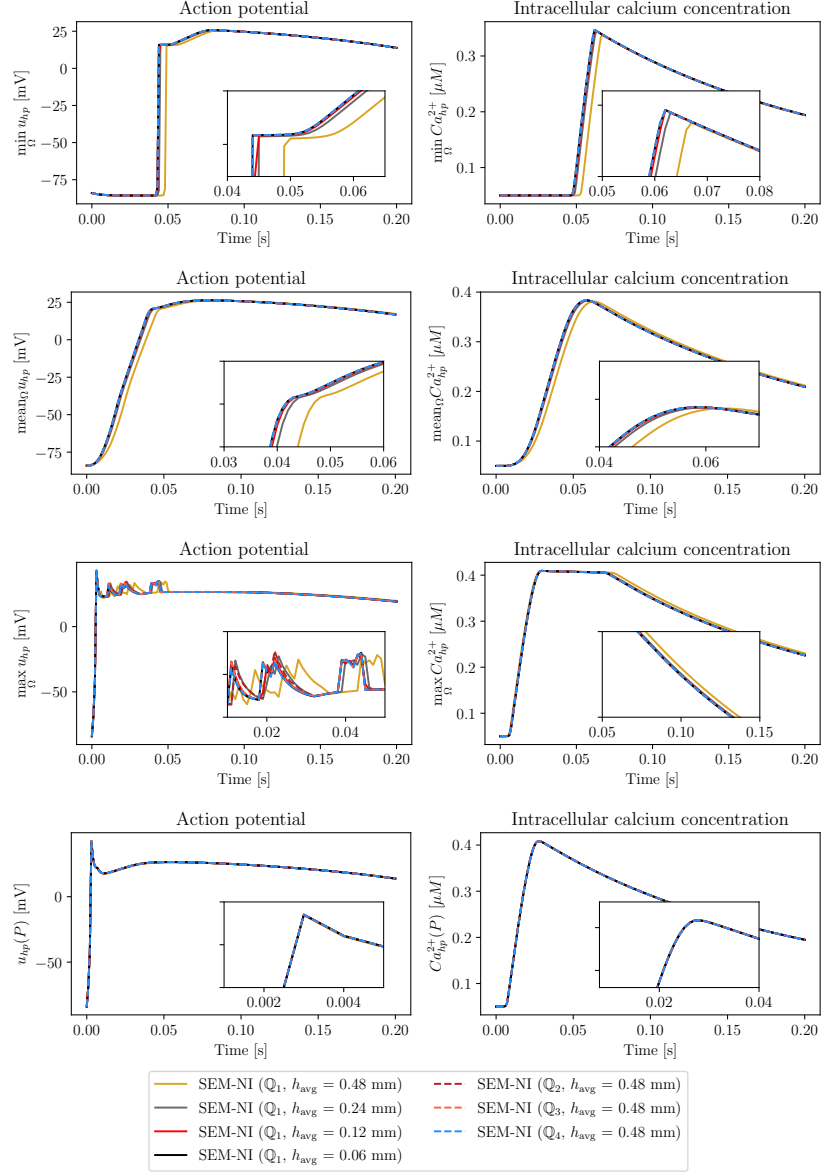
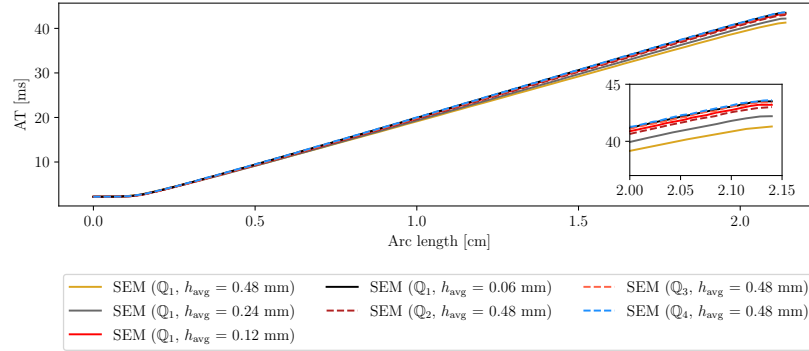
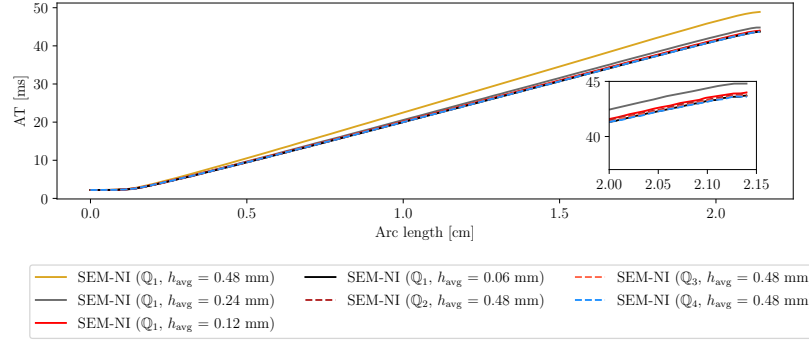


Figure 6: *Niederer benchmark*. Minimum (top), average (second), maximum (third) and point values (bottom) of the action potential u and intracellular calcium concentration Ca^{2+} over time for a slab of cardiac tissue. P is a random point within the computational domain away from the initial stimulus. We consider different hp combinations: SEM-NI \mathbb{Q}_1 to \mathbb{Q}_4 and $h_{\text{avg}} = 0.48$ mm to $h_{\text{avg}} = 0.06$ mm (h_{avg} is the average mesh size).



(a) SEM.



(b) SEM-NI.

Figure 7: *Niederer benchmark*. Activation times computed along the diagonal of the slab (see Figure 4), with different choices of the local space (\mathbb{Q}_1 to \mathbb{Q}_4) and mesh refinements ($h_{\text{avg}} = 0.48$ mm to $h_{\text{avg}} = 0.06$ mm).

accumulates over the diagonal, the inset plots show a zoom around the right endpoint. Such results demonstrate that high polynomial degrees p , even with a coarse mesh size h , lead to a faster convergence rate compared to the small- p , small- h scenario.

To better investigate the comparison between SEM and SEM-NI, in Figure 8

we show the quantities

$$\text{err}_{\max} = \left(\Delta t \sum_n \left| \max_{\mathbf{x}} u_{hp}(\mathbf{x}, t_n) - \max_{\mathbf{x}} u_{\text{ref}}(\mathbf{x}, t_n) \right|^2 \right)^{1/2} \quad (9)$$

$$\text{err}_{\min} = \left(\Delta t \sum_n \left| \min_{\mathbf{x}} u_{hp}(\mathbf{x}, t_n) - \min_{\mathbf{x}} u_{\text{ref}}(\mathbf{x}, t_n) \right|^2 \right)^{1/2} \quad (10)$$

$$\text{err}_{\text{mean}} = \left(\Delta t \sum_n \left| \text{mean}_{\mathbf{x}} u_{hp}(\mathbf{x}, t_n) - \text{mean}_{\mathbf{x}} u_{\text{ref}}(\mathbf{x}, t_n) \right|^2 \right)^{1/2} \quad (11)$$

$$\text{err}_P = \left(\Delta t \sum_n \left| u_{hp}(P, t_n) - u_{\text{ref}}(P, t_n) \right|^2 \right)^{1/2} \quad (12)$$

versus the total number of mesh points, for both SEM and SEM–NI solution u_{hp} . Our reference solution u_{ref} has been computed with \mathbb{Q}_1 –FEM on a very fine grid with average mesh size $h_{\text{avg}} = 0.06$ mm, for a total of 11’401’089 mesh points. P is a random point within the computational domain away from the initial stimulus. The max, min, and mean functions are evaluated on the set of nodes of the mesh. The number of mesh points increases by reducing h for both “SEM h ” and “SEM–NI h ”, while it increases with p for both “SEM p ” and “SEM–NI p ”.

The numerical results confirm the typical behaviour of SEM and SEM–NI discretizations, *i.e.* the errors decrease faster by increasing p rather than decreasing h . Moreover, we notice that SEM and SEM–NI errors behave quite similarly, with a slight advantage for SEM.

In Tables 2–5 we report the CPU time required by the linear solver for the whole numerical simulation (the times are cumulative over all time steps), for SEM and SEM–NI discretizations, matrix–free and matrix–based solvers. Furthermore, in Figure 9 we plot the errors (9)–(12) versus the CPU time required to solve all the linear systems along the whole numerical simulation. For SEM–NI we only report the times relative to the matrix–free solver, while for SEM we report the times for both the matrix–free and matrix–based solvers. The same symbol (circle, square, diamond, and cross) refers to the numerical simulations carried out on the meshes with the same number of DOFs. If we compare the errors and the CPU–times of \mathbb{Q}_1 –SEM, $h_{\text{avg}} = 0.12$ mm with those of \mathbb{Q}_4 –SEM, $h_{\text{avg}} = 0.48$ mm (these two configurations share the same number 1’449’665 of mesh nodes), we notice that the errors of \mathbb{Q}_4 –SEM are at most about $1/3 - 1/2$ of that of \mathbb{Q}_1 –SEM and the ratio of the corresponding CPU times is about 40%. Thus, we conclude that \mathbb{Q}_4 –SEM outperforms \mathbb{Q}_1 –SEM (that is \mathbb{Q}_1 –FEM).

For the comparison between matrix–free and matrix–based solvers, we notice that the former one is always faster, and the gain of matrix–free over matrix–based solver increases with the polynomial degree p . More precisely, the speed–up factors are shown in Table 6 when $h_{\text{avg}} = 0.48$ mm, and in Table 7 when $p = 1$.

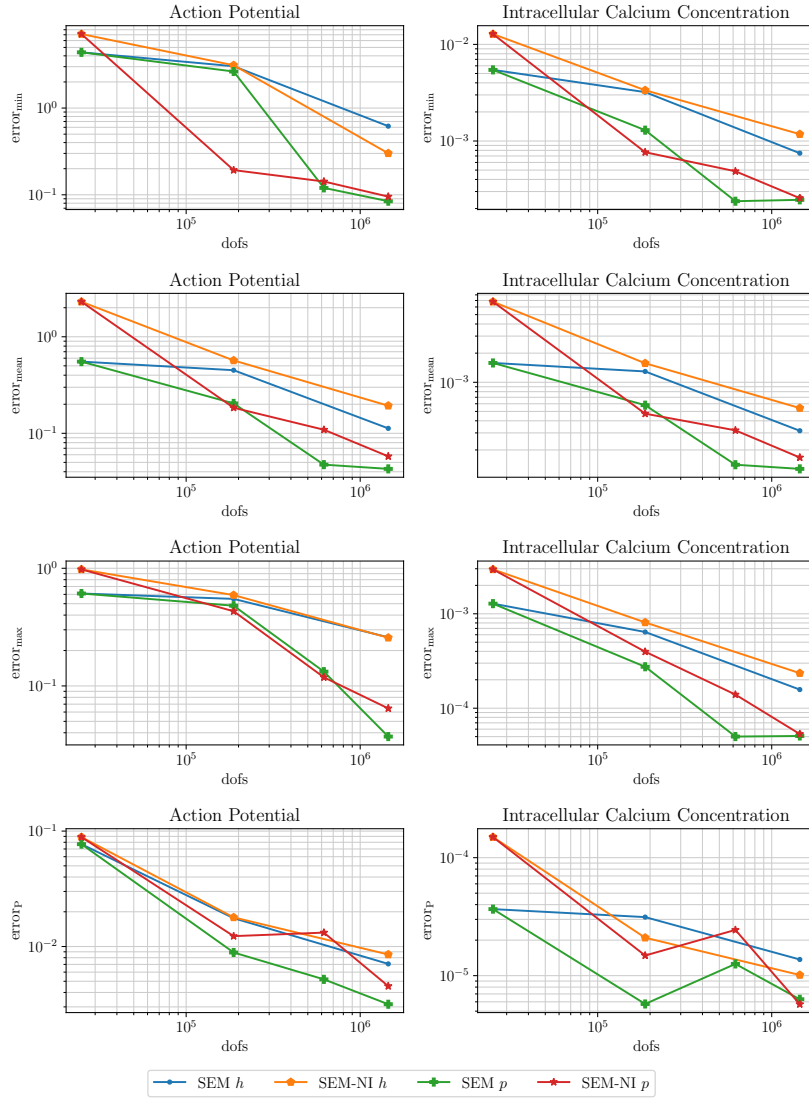


Figure 8: *Niederer benchmark*. Errors for action potential u and intracellular calcium concentration Ca^{2+} versus the number of mesh points (dofs) used in a slab of cardiac tissue.

Mesh points number	Cells number	Local space	Linear solver SEM [s]	Linear solver SEM-NI [s]	Assemble rhs SEM [s]	Assemble rhs SEM-NI [s]
25'025	21'888	Q ₁	10.769	8.031	0.784	0.766
187'425	21'888	Q ₂	14.694	12.383	4.710	4.645
618'529	21'888	Q ₃	37.733	36.419	14.867	14.542
1'449'665	21'888	Q ₄	91.380	90.370	33.899	32.920

Table 2: *Niederer benchmark*. Computational times for SEM and SEM-NI, with a fixed average mesh size $h_{\text{avg}} = 0.48$ mm and p ranging from 1 to 4. Matrix-free solver.

Mesh points number	Cells number	h_{avg} [mm]	Linear solver SEM [s]	Linear solver SEM-NI [s]	Assemble rhs SEM [s]	Assemble rhs SEM-NI [s]
25'025	21'888	0.48	10.769	8.031	0.784	0.766
187'425	175'104	0.24	29.157	27.951	5.771	5.656
1'449'665	1'400'832	0.12	256.295	270.959	42.783	43.548
11'401'089	11'206'656	0.06	2137.329	2158.751	336.272	336.641

Table 3: *Niederer benchmark*. Computational times for SEM and SEM-NI, with an average mesh size h_{avg} ranging from 0.48 mm to 0.06 mm and $p = 1$. Matrix-free solver.

Mesh points number	Cells number	Local space	Linear solver matrix-free [s]	Assembly phase matrix-free [s]	Linear solver matrix-based [s]	Assembly phase matrix-based [s]
25'025	21'888	Q ₁	10.769	0.784	5.570	17.150
187'425	21'888	Q ₂	14.694	4.710	47.655	176.375
618'529	21'888	Q ₃	37.733	14.867	556.041	919.298
1'449'665	21'888	Q ₄	91.380	33.899	2796.981	3598.244

Table 4: *Niederer benchmark*. Computational times for matrix-free and matrix-based solvers, SEM Q_p with a fixed average mesh size $h_{\text{avg}} = 0.48$ mm and p ranging from 1 to 4.

Mesh points number	Cells number	h_{avg} [mm]	Linear solver matrix-free [s]	Assembly phase matrix-free [s]	Linear solver matrix-based [s]	Assembly phase matrix-based [s]
25'025	21'888	0.48	10.769	0.784	5.570	17.150
187'425	175'104	0.24	29.157	5.771	11.331	141.349
1'449'665	1'400'832	0.12	256.295	42.783	138.514	1089.666
11'401'089	11'206'656	0.06	2137.329	336.272	1328.839	8857.580

Table 5: *Niederer benchmark*. Computational times for matrix-free and matrix-based, SEM Q₁ with an average mesh size h_{avg} ranging from 0.48 mm to 0.06 mm.

Local space	Q ₁	Q ₂	Q ₃	Q ₄
$\frac{(\text{CPU time})_{\text{mb}}}{(\text{CPU time})_{\text{mf}}}$	~ 2	~ 7	~ 30	~ 50

Table 6: *Niederer benchmark*. Speed-up of the matrix-free solver over the matrix-based one when $h_{\text{avg}} = 0.48$ mm.

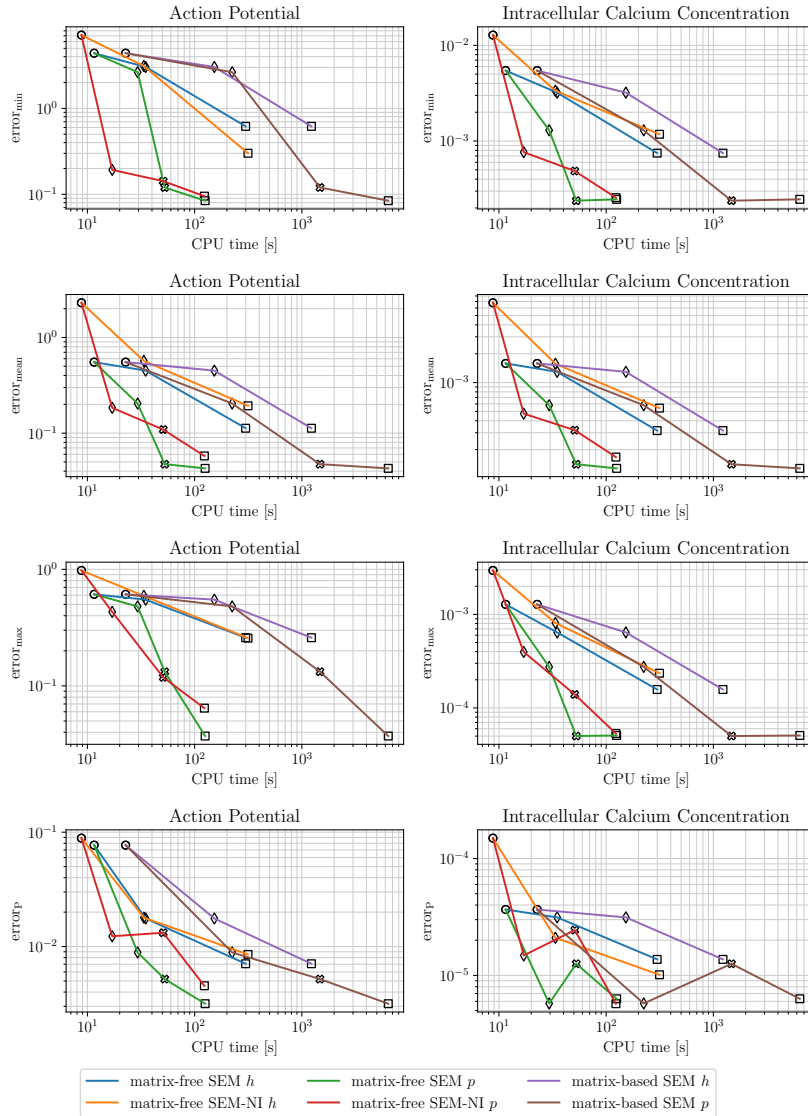


Figure 9: *Niederer benchmark*. Errors of action potential u and intracellular calcium concentration Ca^{2+} versus CPU time required to solve all the linear systems in a slab of cardiac tissue. Equal symbols identify the same number of mesh points: (\circ) 25'025, (\diamond) 187'425, (\times) 618'529, (\square) 1'449'665.

h_{avg}	0.48	0.24	0.12	0.06
$\frac{(\text{CPU time})_{\text{mb}}}{(\text{CPU time})_{\text{mf}}}$	~ 2	~ 4	~ 4	~ 4

Table 7: *Niederer benchmark*. Speed-up of the matrix-free solver over the matrix-based one when $p = 1$.

Solver	Monodomain solver	Monodomain assembly	Ionic model solver
Matrix-based (\mathbb{Q}_1 , $h_{\text{avg}} = 0.12$ mm)	16.61 %	56.98 %	26.41 %
Matrix-free (\mathbb{Q}_4 , $h_{\text{avg}} = 0.48$ mm)	14.95 %	3.36 %	81.69 %

Table 8: *Niederer benchmark*. Matrix-free and matrix-based percentages for the assembling and solving phases. Note that in the matrix-free case we only need to assemble the right-hand side vector, as there is no matrix. Moreover, these percentages are computed without taking into account all other phases of the numerical simulation, such as mesh allocation, fiber generation and output of the results.

Moreover, from Table 8 we observe that, in a matrix-based electrophysiological simulation, most of the computational time is spent to solve the linear system associated with the monodomain equation. On the contrary, in the matrix-free solver most of the computational time is devoted to the ionic model. This means that the cost for solving the linear system has been highly optimized.

Finally, we compare the performance of the AMG and GMG preconditioners, used by the matrix-based and matrix-free solvers, respectively. In Tables 9 and 10 we show the average number of iterations required by the PCG method to solve the linear system (6). We notice that, for the values of h and p considered here, the GMG preconditioner appears optimal in the number of PCG iterations versus both h and p . As a matter of fact, the average number of iterations is about 1.8 for all hp configurations. On the contrary, the AMG preconditioner is optimal versus the mesh size h (even if it requires a slightly larger number of iterations than GMG), while it is not against the polynomial degree p . Indeed, in this case the number of iterations is proportional to the polynomial degree.

All the numerical results shown in this section highlight how much advantageous the matrix-free solver with SEM or SEM-NI is for cardiac electrophysiol-

Mesh points number	Cells number	Local space	Matrix-free (SEM) GMG preconditioner	Matrix-free (SEM-NI) GMG preconditioner	Matrix-based (SEM) AMG preconditioner
25'025	21'888	\mathbb{Q}_1	1.6362	1.8126	2.5777
187'425	21'888	\mathbb{Q}_2	1.8056	1.8581	5.9255
618'529	21'888	\mathbb{Q}_3	1.7906	1.8161	8.3363
1'449'665	21'888	\mathbb{Q}_4	1.7371	1.7826	10.5187

Table 9: *Niederer benchmark*. Average number of CG iterations for matrix-free (SEM, SEM-NI) and matrix-based (SEM) solvers, \mathbb{Q}_p with a fixed average mesh size $h_{\text{avg}} = 0.48$ mm and p ranging from 1 to 4.

Mesh points number	Cells number	h_{avg} [mm]	Matrix-free (SEM) GMG preconditioner	Matrix-free (SEM-NI) GMG preconditioner	Matrix-based (SEM) AMG preconditioner
25'025	21'888	0.48	1.6362	1.8126	2.5777
187'425	175'104	0.24	1.5757	1.6847	3.0535
1'449'665	1'400'832	0.12	1.4448	1.5937	2.9400
11'401'089	11'206'656	0.06	1.4468	1.4923	3.2094

Table 10: *Niederer benchmark*. Average number of CG iterations for matrix-free (SEM, SEM-NI) and matrix-based (SEM) solvers, \mathbb{Q}_1 with an average mesh size h_{avg} ranging from 0.48 mm to 0.06 mm.

Mesh points number	Cells number	Local space	PCG iterations GMG preconditioner
159'149	139'684	\mathbb{Q}_1	2.0770
1'172'919	139'684	\mathbb{Q}_2	1.9628
3'879'415	139'684	\mathbb{Q}_3	1.9455
9'116'741	139'684	\mathbb{Q}_4	1.9440

Table 11: *Zygote left ventricle*. Average number of PCG iterations for the matrix-free solver with SEM discretization, \mathbb{Q}_p with a fixed average mesh size $h_{\text{avg}} = 2.0$ mm and p ranging from 1 to 4.

ogy simulations, with respect to the matrix-based solver with low-order FEM.

Since the matrix-free implementation outperforms the matrix-based one, while SEM and SEM-NI provide comparable results in terms of accuracy and efficiency, we will employ the matrix-free solver with just the SEM formulation for the numerical simulations that we are going to present in the next sections.

5.2. Left ventricle

We report the results for the electrophysiological simulations performed with the Zygote left ventricle geometry. The settings of this test case are resumed at the beginning of Sect. 5. We consider a mesh with $h_{\text{avg}} = 2.0$ mm and polynomial degree p from 1 to 4.

In Table 11 we report the number of mesh nodes, the number of cells and the average number of iterations required by the PCG method to solve the linear system (6). As for the Niederer benchmark, the GMG preconditioner turns out to be optimal also for these numerical simulations. Indeed, the number of PCG iterations is about 2 along the whole time history for any polynomial degree between 1 and 4.

In Figure 10 we depict the activation maps for different choices of the local space (from \mathbb{Q}_1 to \mathbb{Q}_4). By looking at the isolines, we observe that the \mathbb{Q}_3 solution is very close to the \mathbb{Q}_4 solution, that means we reach convergence for $p = 3$, even with such a relatively low mesh resolution $h_{\text{avg}} = 2.0$ mm. Whereas, it is a well-established result in the literature that first order Finite Elements would reach convergence for a value of h_{avg} that is about 100 times smaller – *i.e.* for a much higher number of DOFs (see, *e.g.*, [14]). DOFs refer to the degrees of freedom of the action potential, disregarding both gating variables and ionic species, then it coincides with the number of mesh nodes.

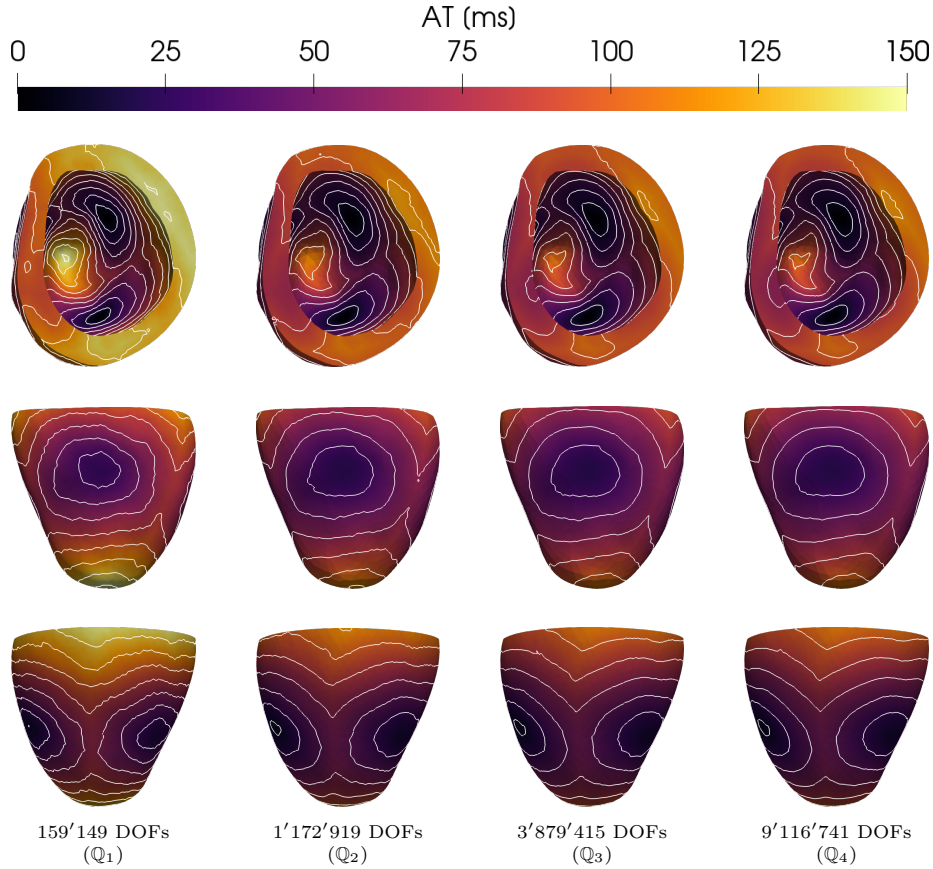


Figure 10: *Zygote left ventricle*. Three views of the activation maps computed with \mathbb{Q}_p elements ($p = 1, \dots, 4$) and a fixed average mesh size $h_{\text{avg}} = 2.0$ mm.

The same conclusions hold when considering Figure 11, where we show the minimum, average, and maximum pointwise values of both the action potential u and the intracellular calcium concentration Ca^{2+} over time.

5.3. Whole-heart

The aim of this section is to show that our matrix-free solver can be successfully applied even in a complex framework. For this purpose we perform a numerical simulation in sinus rhythm with the Zygote four-chamber heart. The settings of this test case can be found at the beginning of Sect. 5. We consider different ionic models, namely CRN and TTP06 for atria and ventricles, respectively. Furthermore, we model the valvular rings as non-conductive regions of the myocardium. The mesh is endowed with 355'664 cells and 10'355'058 nodes ($h_{\text{avg}} = 1.6$ mm). We employ the matrix-free solver and \mathbb{Q}_3 -SEM, this choice is motivated by the numerical results obtained for the Niederer bench-

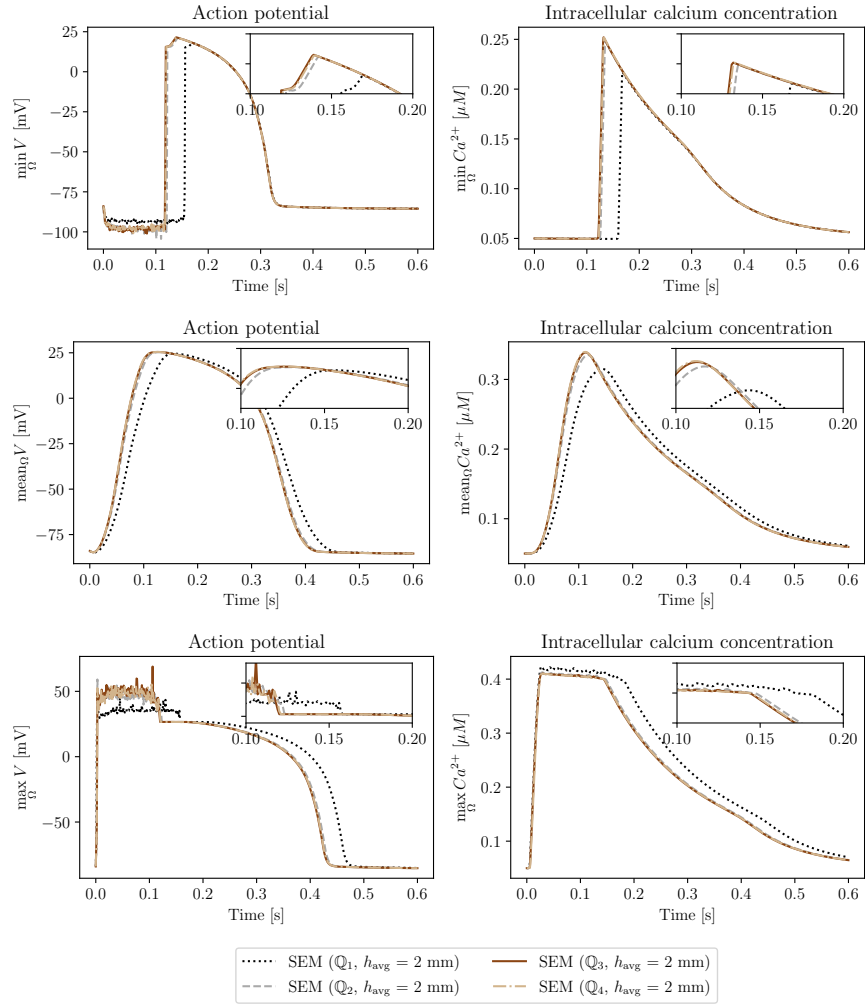


Figure 11: *Zygote left ventricle*. Minimum (top), average (mid), maximum (bottom) pointwise values of action potential u and intracellular calcium concentration Ca^{2+} over time, \mathbb{Q}_p with a fixed average mesh size $h_{\text{avg}} = 2.0$ mm and p ranging from 1 to 4.

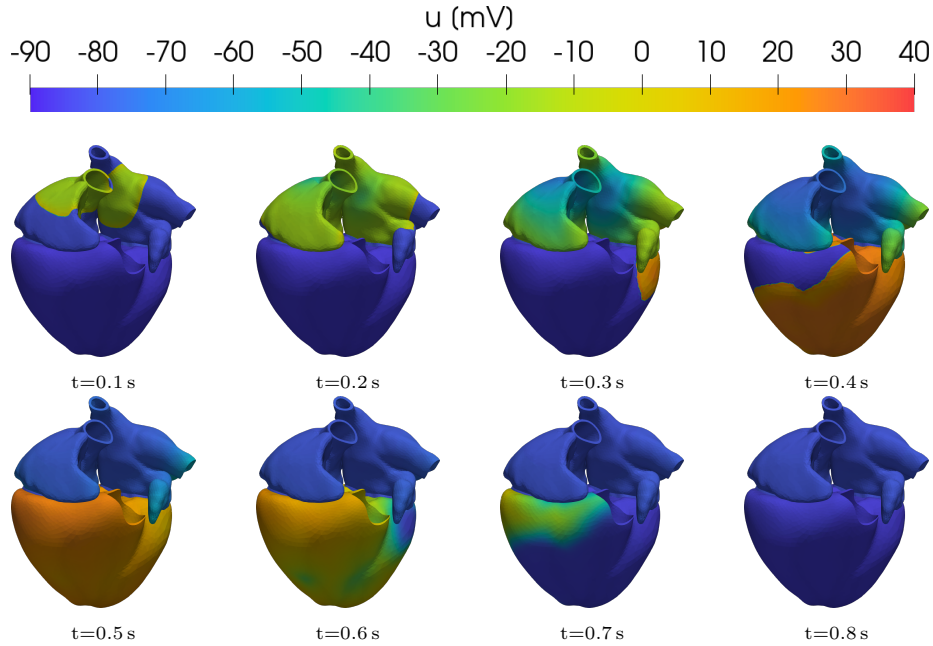


Figure 12: *Zygote whole-heart*. Time evolution of the transmembrane potential u during one heartbeat.

mark (Sect. 5.1) and the convergence test performed on the Zygote left ventricle geometry (Sect. 5.2).

We depict in Figure 12 the evolution of the transmembrane potential over time on the whole-heart geometry. The electric signal initiates at the sinoatrial node in the right atrium and then propagates to the left atrium and ventricles by means of preferential conduction lines, such as the Bachmann’s and His bundles [60]. The wavefront propagation appears very smooth, while accounting for small dissipation and dispersion throughout the heartbeat [22].

6. Conclusions

We developed a matrix-free solver for cardiac electrophysiology tailored to the efficient use of high-order numerical methods. We employed the monodomain equation to model the propagation of the transmembrane potential and physiologically-based ionic models (CRN and TTP06) to describe the behaviour of different chemical species at the cells level.

We run several electrophysiological simulations for three different test cases, namely a slab of cardiac tissue, the Zygote left ventricle and the Zygote whole-heart to demonstrate the effectiveness and generality of our matrix-free solver in combination with Spectral Element Methods. SEM and SEM-NI provided comparable numerical results in terms of both accuracy and efficiency. Further-

more, we showed the importance of considering high-order Finite Elements for this class of mathematical problems, *i.e.* when sharp wavefronts are involved.

Our matrix-free solver outperforms state-of-the-art matrix-based solvers in terms of computational costs and memory requirements. This is true even when matrix-vector products are computed without any matrix-free preconditioner, thanks to both vectorization and sum-factorization. Finally, the small memory usage of the matrix-free implementation may allow for the development of GPU-based solvers of the cardiac function.

Acknowledgments

This research has been funded partly by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 740132, iHEART “An Integrated Heart Model for the simulation of the cardiac function”, P.I. Prof. A. Quarteroni) and partly by the Italian Ministry of University and Research (MIUR) within the PRIN (Research projects of relevant national interest 2017 “Modeling the heart across the scales: from cardiac cells to the whole organ” Grant Registration number 2017AXL54F).

Authors’ contribution

All authors contributed to the study conception and design of the work. PCA and MS developed the computer code, performed the numerical simulations and post-processed the numerical results. PG conceptualized the goals and the theoretical background of the research and took the lead in designing the experiments and interpreting the results. LD contributed to the analysis of results and coordinated the research activity planning. The initial draft of this manuscript was written by PCA, MS and PG. All authors read and approved this manuscript.

References

- [1] T. Gerach, S. Schuler, J. Fröhlich, L. Lindner, , E. Kovacheva, R. Moss, E. Wülfers, G. Seemann, C. Wieners, A. Loewe, Electro-Mechanical Whole-Heart Digital Twins: A Fully Coupled Multi-Physics Approach, *Mathematics* 9 (11).
- [2] R. Gray, P. Pathmanathan, Patient-Specific Cardiovascular Computational Modeling: Diversity of Personalization and Challenges, *Journal of Cardiovascular Translational Research* 11 (2018) 80–88.
- [3] R. Piersanti, F. Regazzoni, M. Salvador, A. Corno, L. Dede’, C. Vergara, A. Quarteroni, 3D-0D closed-loop model for the simulation of cardiac biventricular electromechanics, *Computer Methods in Applied Mechanics and Engineering* 391 (2022) 114607.

- [4] M. Potse, D. Krause, W. Kroon, R. Murzilli, S. Muzzarelli, F. Regoli, E. Caiani, F. Prinzen, R. Krause, A. Auricchio, Patient-specific modelling of cardiac electrophysiology in heart-failure patients, *Europace* 16 (2014) v56–iv61.
- [5] M. Strocchi, C. Augustin, M. Gsell, E. Karabelas, A. Neic, K. Gillette, O. Razeghi, A. Prassl, E. Vigmond, J. Behar, J. Gould, B. Sidhu, C. Rinaldi, M. Bishop, G. Plank, S. Niederer, A publicly available virtual cohort of four-chamber heart meshes for cardiac electro-mechanics simulations, *PLOS ONE* 15 (2020) 1–26.
- [6] A. Quarteroni, L. Dedè, A. Manzoni, C. Vergara, *Mathematical Modelling of the Human Cardiovascular System: Data, Numerical Approximation, Clinical Applications*, Cambridge University Press, 2019.
- [7] N. A. Trayanova, R. Winslow, Whole-Heart Modeling: Applications to Cardiac Electrophysiology and Electromechanics, *Circulation Research* 108 (1) (2011) 113–128.
- [8] P. Colli Franzone, L. Pavarino, S. Scacchi, *Mathematical cardiac electrophysiology*, Vol. 13, Springer, 2014.
- [9] H. Arevalo, F. Vadakkumpadan, E. Guallar, A. Jebb, P. Malamas, K. Wu, N. Trayanova, Arrhythmia risk stratification of patients after myocardial infarction using personalized heart models, *Nature Communications* 7 (2016) 11437.
- [10] J. Bayer, V. Sobota, A. Moreno, P. Jais, E. Vigmond, The Purkinje network plays a major role in low-energy ventricular defibrillation, *Computers in Biology and Medicine* 141 (2022) 105133.
- [11] K. Gillette, M. Gsell, A. Prassl, E. Karabelas, U. Reiter, G. Reiter, T. Grandits, C. Payer, D. Štern, M. Urschler, J. Bayer, C. Augustin, A. Neic, T. Pock, E. Vigmond, G. Plank, A Framework for the generation of digital twins of cardiac electrophysiology from clinical 12-leads ECGs, *Medical Image Analysis* 71 (2021) 102080.
- [12] C. Mendonca Costa, A. Neic, E. Kerfoot, B. Porter, B. Sieniewicz, J. Gould, B. Sidhu, Z. Chen, G. Plank, C. Rinaldi, M. Bishop, S. Niederer, Pacing in proximity to scar during cardiac resynchronization therapy increases local dispersion of repolarization and susceptibility to ventricular arrhythmogenesis, *Heart Rhythm* 16 (10) (2019) 1475–1483.
- [13] A. Quarteroni, A. Valli, *Numerical Approximation of Partial Differential Equations*, Springer Verlag, Heidelberg, 1994.
- [14] L. Woodworth, B. Cansız, M. Kaliske, A numerical study on the effects of spatial and temporal discretization in cardiac electrophysiology, *International Journal for Numerical Methods in Biomedical Engineering* 37 (5) (2021) e3443.

- [15] A. Patera, A spectral element method for fluid dynamics: laminar flow in a channel expansion, *Journal of Computational Physics* 54 (1984) 468–488.
- [16] Y. Maday, A. Patera, Spectral element methods for the incompressible Navier-Stokes equations, in: *State-of-the-Art Surveys on Computational Mechanics*, A.K. Noor and J. T. Oden, 1989, pp. 71–143.
- [17] C. Canuto, M. Hussaini, A. Quarteroni, T. Zang, *Spectral Methods. Evolution to Complex Geometries and Applications to Fluid Dynamics*, Springer, Heidelberg, 2007.
- [18] D. Arnold, F. Brezzi, B. Cockburn, L. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, *SIAM Journal on Numerical Analysis* 39 (5) (2001) 1749–1779.
- [19] B. Cockburn, C.-W. Shu, The local discontinuous galerkin method for time-dependent convection-diffusion systems, *SIAM Journal on Numerical Analysis* 35 (6) (1998) 2440–2463.
- [20] R. LeVeque, *Finite volume methods for hyperbolic problems*, Cambridge Texts in Applied Mathematics, Cambridge University Press, Cambridge, 2002.
- [21] J. A. Cottrell, T. J. R. Hughes, Y. Bazilevs, *Isogeometric Analysis: Toward Integration of CAD and FEA*, Wiley, 2009.
- [22] M. Bucelli, M. Salvador, L. Dedè, A. Quarteroni, Multipatch Isogeometric Analysis for electrophysiology: Simulation in a human heart, *Computer Methods in Applied Mechanics and Engineering* 376 (2021) 113666.
- [23] C. Cantwell, S. Yakovlev, R. Kirby, N. Peters, S. Sherwin, High-order spectral/hp element discretisation for reaction–diffusion problems on surfaces: Application to cardiac electrophysiology, *Journal of Computational Physics* 257 (2014) 813–829.
- [24] Y. Coudière, R. Turpault, Very high order finite volume methods for cardiac electrophysiology, *Computers & Mathematics with Applications* 74 (4) (2017) 684–700.
- [25] J. Hoermann, C. Bertoglio, M. Kronbichler, M. Pfaller, R. Chabiniok, W. Wall, An adaptive hybridizable discontinuous Galerkin approach for cardiac electrophysiology, *International Journal for Numerical Methods in Biomedical Engineering* 34 (5).
- [26] K. Vincent, M. Gonzales, A. Gillette, C. Villongco, S. Pezzuto, J. Omens, M. Holst, A. McCulloch, High-order finite element methods for cardiac monodomain simulations, *Frontiers in Physiology* 6 (Aug).

- [27] D. Arndt, N. Fehn, G. Kanschat, K. Kormann, M. Kronbichler, P. Munch, W. Wall, J. Witte, ExaDG: High-Order Discontinuous Galerkin for the Exa-Scale, in: Software for Exascale Computing - SPPEXA 2016-2019, Springer International Publishing, Cham, 2020, pp. 189–224.
- [28] S. Orszag, Spectral methods for problem in complex geometries, Journal of Computational Physics 37 (1980) 70–92.
- [29] J. Melenk, K. Gerdes, C. Schwab, Fully discrete *hp*-finite elements: Fast quadrature, Computer Methods in Applied Mechanics and Engineering 190 (32-33) (2001) 4339 – 4364.
- [30] M. Kronbichler, K. Kormann, A generic interface for parallel cell-based finite element operator application, Computers and Fluids 63 (2012) 135–147.
- [31] Y. Xia, K. Wang, H. Zhang, Parallel Optimization of 3D Cardiac Electrophysiological Model Using GPU, Computational and Mathematical Methods in Medicine 2015 (2015) 862735.
- [32] M. Kronbichler, K. Ljungkvist, Multigrid for Matrix-Free High-Order Finite Element Computations on Graphics Processors, ACM Transactions on Parallel Computing 6 (1) (2019) 1–32.
- [33] G. Del Corso, R. Verzicco, F. Viola, A fast computational model for the electrophysiology of the whole human heart, Journal of Computational Physics 457 (2022) 111084.
- [34] 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, N. Smith, Verification of cardiac tissue electrophysiology simulators using an N-version benchmark, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 369 (1954) (2011) 4331–51.
- [35] D. Arndt, W. Bangerth, B. Blais, T. Clevenger, M. Fehling, A. Grayver, T. Heister, L. Heltai, M. Kronbichler, M. Maier, P. Munch, J. Pelteret, R. Rastak, I. Tomas, B. Turcksin, Z. Wang, D. Wells, The deal.II library, Version 9.2, Journal of Numerical Mathematics 28 (3) (2020) 131–146.
- [36] K. ten Tusscher, A. Panfilov, Alternans and spiral breakup in a human ventricular tissue model, American Journal of Physiology Heart and Circulation Physiology 291 (2006) 1088–1100.
- [37] M. Courtemanche, R. Ramirez, S. Nattel, Ionic mechanisms underlying human atrial action potential properties: insights from a mathematical model, American Journal of Physiology Heart and Circulation Physiology 275 (1) (1998) H301–H321.

- [38] R. Piersanti, P. Africa, M. Fedele, C. Vergara, L. Dedè, A. Corno, A. Quarteroni, Modeling cardiac muscle fibers in ventricular and atrial electrophysiology simulations, *Computer Methods in Applied Mechanics and Engineering* 373 (2021) 113468.
- [39] Zygote Media Group Inc., Zygote Solid 3D heart Generation II, Development Report (2014).
- [40] C. Bernardi, Y. Maday, Spectral, spectral element and mortar element methods, in: *Theory and numerics of differential equations* (Durham, 2000), Universitext, Springer, Berlin, 2001, pp. 1–57.
- [41] G. Karniadakis, S. Sherwin, *Spectral/hp Element Methods for Computational Fluid Dynamics*, Oxford University Press, 2005, 2nd ed.
- [42] C. Canuto, M. Hussaini, A. Quarteroni, T. Zang, *Spectral Methods. Fundamentals in Single Domains*, Springer, Heidelberg, 2006.
- [43] B. Szabó, I. Babuška, *Finite Element Analysis*, John Wiley & sons, New York, 1991.
- [44] C. Schwab, *p - and hp - finite element methods*, Oxford University Press, Oxford, 1998.
- [45] N. Fehn, W. Wall, M. Kronbichler, A matrix-free high-order discontinuous Galerkin compressible Navier-Stokes solver: A performance comparison of compressible and incompressible formulations for turbulent incompressible flows, *International Journal for Numerical Methods in Fluids* 89 (3) (2019) 71–102.
- [46] F. Regazzoni, M. Salvador, P. Africa, M. Fedele, L. Dede', A. Quarteroni, A cardiac electromechanical model coupled with a lumped-parameter model for closed-loop blood circulation, *Journal of Computational Physics* 457 (2022) 111083.
- [47] A. Quarteroni, T. Lassila, S. Rossi, R. Ruiz-Baier, Integrated Heart-Coupling multiscale and multiphysics models for the simulation of the cardiac function, *Computer Methods in Applied Mechanics and Engineering* 314 (2017) 345–407.
- [48] P. Gervasio, F. Saleri, A. Veneziani, Algebraic fractional step schemes with spectral methods for the incompressible Navier-Stokes equations, *Journal of Computational Physics* 214 (1) (2006) 347–365.
- [49] J. M. Cebrian, L. Natvig, M. Jahre, Scalability analysis of AVX-512 extensions, *The Journal of Supercomputing* 76 (3) (2020) 2082–2097.
- [50] D. Zhong, Q. Cao, G. Bosilca, J. Dongarra, Using long vector extensions for MPI reductions, *Parallel Computing* 109 (2022) 102871.

- [51] M. Kronbichler, K. Kormann, Fast matrix-free evaluation of discontinuous Galerkin finite element operators, *ACM Transactions on Mathematical Software* 45 (3) (2019) 1–40.
- [52] C. Cantwell, S. Sherwin, R. Kirby, P. Kelly, From h to p efficiently: Strategy selection for operator evaluation on hexahedral and tetrahedral elements, *Computers and Fluids* 43 (1) (2011) 23 – 28.
- [53] B. Janssen, G. Kanschat, Adaptive Multilevel Methods with Local Smoothing for H^1 - and H^{curl} -Conforming High Order Finite Element Methods, *SIAM Journal on Scientific Computing* 33 (4) (2011) 2095–2114.
- [54] J. Xu, L. Zikatanov, Algebraic multigrid methods, *Acta Numerica* 26 (2017) 591 – 721.
- [55] H. Sundar, G. Stadler, G. Biros, Comparison of multigrid algorithms for high-order continuous finite element discretizations, *Numerical Linear Algebra with Applications* 22.
- [56] U. Trottenberg, C. Oosterlee, A. Schüller, *Multigrid*, Elsevier Academic Press, London, UK, 2001.
- [57] T. Clevenger, T. Heister, G. Kanschat, M. Kronbichler, A Flexible, Parallel, Adaptive Geometric Multigrid Method for FEM, *ACM Transactions on Mathematical Software* 47 (1).
- [58] M. Adams, M. Brezina, J. Hu, R. Tuminaro, Parallel multigrid smoothing: Polynomial versus Gauss-Seidel, *Journal of Computational Physics* 188 (2) (2003) 593 – 610.
- [59] P. Africa, R. Piersanti, M. Fedele, L. Dedè, A. Quarteroni, lifex – heart module: a high-performance simulator for the cardiac function. Package 1: Fiber generation, arXiv preprint arXiv:2108.01907.
- [60] R. Harrington, J. Narula, Z. Eapen, *Hurst’s the Heart*, MacGraw-Hill, 2011.