



POLITECNICO
MILANO 1863

COURSE OF NUMERICAL ANALYSIS FOR PARTIAL DIFFERENTIAL EQUATIONS

A HIGH-ORDER DISCONTINUOUS GALERKIN METHOD FOR THE BIDOMAIN PROBLEM OF CARDIAC ELECTROPHYSIOLOGY

Authors: FEDERICA BOTTA, MATTEO CALAFÀ

Supervisors: CHRISTIAN VERGARA, PAOLA ANTONIETTI

A.Y. 2020/2021

Contents

1	Introduction	3
1.1	The physical problem	3
1.2	Mathematical models	4
1.3	State of the art	5
2	Semi-discrete Discontinuous Galerkin formulation	6
2.1	Discontinuous Galerkin discrete formulation	6
2.2	Algebraic formulation	7
3	Implementation details: the Dubiner Basis	9
3.1	Analytical aspects	9
3.2	Implementation	12
3.3	Switch from the modal expansion coefficients to the Lagrangian ones	12
3.3.1	From Dubiner Basis to FEM Basis	13
3.3.2	From FEM Basis to Dubiner Basis	13
3.3.3	Final remarks	13
4	Temporal discretization	15
4.1	The Semi-implicit method	15
4.2	The Godunov operator-splitting	16
4.3	The Quasi-implicit operator-splitting	17
5	About uniqueness of the potentials	19
5.1	Analytical concepts	19
5.2	Numerical correction	20
5.2.1	Implementation of the first coefficient imposition	22
5.2.2	An analytical motivation for the mean value imposition method	23
5.2.3	Implementation of the mean value imposition	27
5.2.4	Final remarks	28
6	Error analysis results	29
6.1	Data	29
6.2	Comparison between Dubiner and FEM	29
6.2.1	P1-D1	29
6.2.2	P2-D2	31
6.2.3	P3-D3	33
6.3	Comparison between different time discretization methods	35
6.4	Comparison between methods for uniqueness of ϕ_i and ϕ_e	38
7	Realistic simulations	40
7.1	First test-case	41
7.2	Second test-case	44

8 Numerical Codes	47
8.1 Implementation: Dubiner to FEM	47
8.2 Implementation: FEM to Dubiner	47
8.3 Implementation: Semi-implicit method	48
8.4 Implementation: Godunov operator-splitting method	48
8.5 Implementation: Quasi-implicit operator-splitting method	49
8.6 Implementation: imposing a value in a specific point	49
8.7 Implementation: imposing zero mean	50

1 Introduction

The aim of this project is to study and implement a suitable numerical scheme for the approximate solutions of the *Bidomain Problem*, a well-known system of non-linear partial differential equations that has been developed in the context of the modelling of electrophysiology of human heart.

This work is basically the continuation of a two-years-long study carried out by three past course projects ([4], [2], [9]). In particular, the broad goal of this project is to improve the results obtained in [9] (Marta and Perego) for the Bidomain model. Indeed, we provide further analysis to go deeper in the study of stability and thus convergence of a *Discontinuous Galerkin* discretization and to assess the reliability of this method for more realistic scenarios. As a matter of fact this work is primarily based on these provided data and MATLAB[©] codes.

1.1 The physical problem

We provide in what follows a brief introduction to the Bidomain equations. For a more complete explanation, we instead refer to [11].

The mechanical contraction and expansion of the human heart has its origin in the *electrical activation* of the cardiac cells. At each heartbeat, myocytes are activated and deactivated following a characteristic electrical cycle (Fig. (1)).

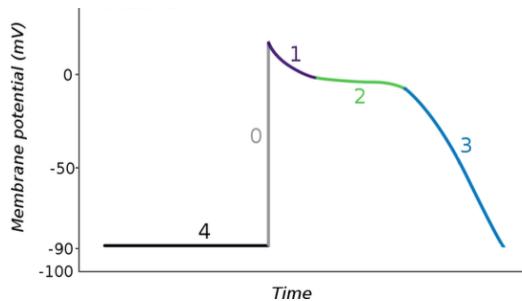


Figure 1: Membrane potential in function of time (one cardiac cycle)

The cell is initially at rest (-90mV , step 4). At a certain point, its potential increases rapidly ($\approx 2\text{ms}$) and reaches the value of $+20\text{mV}$: the cell is activated. Later, a plateau near 0mV is observed and then a slow repolarization to the initial potential; cf. Fig. (1).

From a microscopical point of view, we could study the dynamics acting in each single cell (as a consequence of the passage of chemical ions through specific channels, e.g. $\text{Ca}^{2+}, \text{Na}^+, \text{K}^+$). From a macroscopical point of view, instead, one can describe it as a continuous electrical diffusion over the entire cardiac surface. Even if this consists in a very rapid phenomenon, the study of such propagation could be very interesting in order, for instance, to detect diseases in sick patients.

1.2 Mathematical models

Starting from the circuit in Figure (2), applying general electromagnetism laws, the Bidomain model has been formulated (see [11] for more details and/or [6] for the complete derivation).

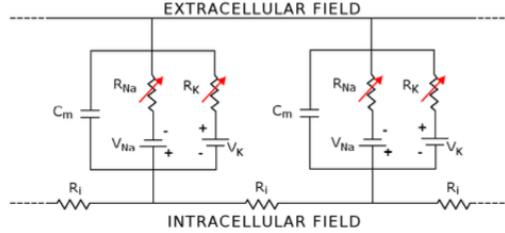


Figure 2: Simplified circuit to model the intracellular and extracellular potential dynamics

The general formulation is the following:

Problem 1 (Bidomain model). *Find ϕ_i and ϕ_e such that:*

$$\begin{cases} \chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_i \nabla \phi_i) + \chi_m I_{ion} = I_i^{ext}, & \text{in } \Omega_{mus} \times (0, T], \\ -\chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_e \nabla \phi_e) - \chi_m I_{ion} = -I_e^{ext}, & \text{in } \Omega_{mus} \times (0, T]. \end{cases}$$

where:

- ϕ_i, ϕ_e are the *Intracellular and Extracellular Potentials* (unknowns),
- $V_m = \phi_i - \phi_e$ is the *Trans-membrane Potential*,
- χ_m, C_m are known positive constants,
- Σ_i, Σ_e are known positive definite tensors,
- I_i^{ext}, I_e^{ext} are applied currents,
- I_{ion} is the *Ionic Current*,
- Ω_{mus} is the cardiac domain (myocardium + endocardium + epicardium).

Actually, this system is not complete since it misses boundary and initial conditions and a suitable model for I_{ion} . Initial conditions and Neumann boundary conditions for ϕ_i and ϕ_e are then imposed. For the definition of I_{ion} , instead, a *reduced ionic model* is chosen, in particular the *FitzHugh-Nagumo model*. Summing up:

Problem 2. [Bidomain + FitzHugh-Nagumo model with Neumann boundary conditions]
Find ϕ_i and ϕ_e such that:

$$\begin{cases} \chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_i \nabla \phi_i) + \chi_m I_{ion}(V_m, w) = I_i^{ext}, & \text{in } \Omega_{mus} \times (0, T], \\ -\chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_e \nabla \phi_e) - \chi_m I_{ion}(V_m, w) = -I_e^{ext}, & \text{in } \Omega_{mus} \times (0, T], \\ I_{ion}(V_m, w) = k V_m (V_m - a)(V_m - 1) + w, & \text{in } \Omega_{mus} \times (0, T], \\ \frac{\partial w}{\partial t} = \epsilon (V_m - \gamma w), & \text{in } \Omega_{mus} \times (0, T], \\ \Sigma_i \nabla \phi_i \cdot n = b_i, & \text{on } \partial \Omega_{mus} \times (0, T], \\ \Sigma_e \nabla \phi_e \cdot n = b_e, & \text{on } \partial \Omega_{mus} \times (0, T], \\ \text{Initial conditions for } \phi_i, \phi_e, w, & \text{in } \Omega_{mus} \times \{t = 0\}. \end{cases}$$

where:

- w is the *gating variable* (unknown),
- k, a, ϵ, γ are known constants,
- b_i, b_e are the boundary conditions data,
- n is the outward normal vector.

From now on, the system of equations given by Problem (2) will be the reference analytical problem for the development of the forthcoming numerical schemes.

To conclude, there exist other famous and useful models, such as the *Monodomain model*. But this is just a simplification of the Bidomain as in this case it is assumed that ϕ_i and ϕ_e are proportional. However, thanks to its simplicity, we often tested the code starting from the Monodomain implementation of the project [2] instead of analyzing directly the Bidomain model reported.

1.3 State of the art

As we have already introduced, our project initially aimed to continue and improve the work of a previous project [9].

Results obtained using unitary parameters, namely $\chi_m = \Sigma_i = \Sigma_e = C_m = k = \epsilon = \gamma = a = 1$, were actually quite satisfactory. On the other hand, the choice of more realistic/experimental values for the parameters (that are often very big or very small) caused bad consequences to the accuracy of the schemes or even to their stability. In particular, we observed that the choice of $C_m \approx 10^{-2}$ highly compromised the stability of the numerical schemes (fact that was already noticed in [2]). This issue heavily limits the use of the MATLAB code for research and/or experimental simulations as it guarantees convergence to the right solution only in few and non-realistic problems.

2 Semi-discrete Discontinuous Galerkin formulation

2.1 Discontinuous Galerkin discrete formulation

Starting from the strong form given by the Problem (2), the next step is the achievement of a suitable Discontinuous Galerkin semi-discrete formulation. Full descriptions and justifications of all the terms are present in [9].

Let us introduce a triangulation τ_h over Ω , where $\mathcal{F}_h = \mathcal{F}_h^I \cup \mathcal{F}_h^B$ on the set of the faces of the partition, which includes the internal (\mathcal{F}_h^I) and boundary (\mathcal{F}_h^B) faces, respectively. Let the DG space be defined as $V_h^p = \{v_h \in L^2(\Omega) : v_h|_{\mathcal{K}} \in \mathbb{P}^p(\mathcal{K}) \quad \forall \mathcal{K} \in \tau_h\}$, where p is the degree of the piecewise continuous polynomial, i.e. $k \geq i$. Moreover, we define $N_h = \dim(V_h^p) < \infty$.

Problem 3 (DG weak formulation). *For any $t \in [0, T]$ find $\Phi_h(t) = [\phi_i^h(t), \phi_e^h(t)]^T \in [V_h^p]^2$ and $w_h(t) \in V_h^p$ such that:*

$$\left\{ \begin{array}{l} \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m C_m \frac{\partial V_m^h}{\partial t} V_h dw + a_i(\phi_i^h, v_h) + \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m k(V_m^h - 1)(V_m^h - a) V_m^h v_h dw + \\ \quad + \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m w_h v_h dw = (I_i^{ext}, v_h), \quad \forall v_h \in V_h^p, \\ \\ - \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m C_m \frac{\partial V_m^h}{\partial t} V_h dw + a_e(\phi_e^h, v_h) - \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m k(V_m^h - 1)(V_m^h - a) V_m^h v_h dw + \\ \quad - \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m w_h v_h dw = (-I_e^{ext}, v_h), \quad \forall v_h \in V_h^p, \\ \\ \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \frac{\partial w_h}{\partial t} v_h dw = \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \epsilon(V_m^h - \gamma w_h) v_h dw, \quad \forall v_h \in V_h^p, \end{array} \right.$$

where:

- $a_l(\phi_l^h, v_h) = \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} (\Sigma_l \nabla_h \phi_k^h) \cdot \nabla_h v_h dw - \sum_{F \in \mathcal{F}_h^I} \int_F \{\{\Sigma_l \nabla_h \phi_l^h\}\} \cdot [[v_h]] d\sigma +$
 $- \delta \sum_{F \in \mathcal{F}_h^I} \int_F \{\{\Sigma_l \nabla_h v_h\}\} \cdot [[\phi_l^h]] d\sigma + \sum_{F \in \mathcal{F}_h^I} \int_F \Gamma[[\phi_l^h]] \cdot [[v_h]] d\sigma \quad l = i, e,$ (1)
- $(I_i^{ext}, v_h) = \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} I_i^{ext} v_h dw + \int_{\partial\Omega} b_i v_h d\sigma,$
- $(-I_e^{ext}, v_h) = - \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} I_e^{ext} v_h dw + \int_{\partial\Omega} b_e v_h d\sigma.$

Moreover, according to the choice of the coefficient δ , we can define:

- $\delta = 1$: Symmetric Interior Penalty method (SIP)
- $\delta = 0$: Incomplete Interior Penalty method (IIP)
- $\delta = -1$: Non Symmetric Interior Penalty method (NIP)

In (1) Γ is the so called stability parameter, which is defined edge-wise as:
 $\Gamma := \alpha \frac{k^2}{h}$, where $\alpha \in \mathbb{R}$ has to be chosen high enough for the SIP and IIP formulations.

2.2 Algebraic formulation

Taking $\{\varphi_j\}_{j=1}^{N_h}$ a basis of V_h^n , so that we can write

$$\begin{aligned}\Phi_h(t) &= \begin{bmatrix} \phi_i^h(t) \\ \phi_e^h(t) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N_h} \phi_{i,j}(t) \varphi_j \\ \sum_{j=1}^{N_h} \phi_{e,j}(t) \varphi_j \end{bmatrix}, \\ w_h(t) &= \sum_{j=1}^{N_h} w_j(t) \varphi_j, \\ V_m^h(t) &= \sum_{j=1}^{N_h} V_{m,j}(t) \phi_j = \sum_{j=1}^{N_h} (\phi_{i,j}(t) - \phi_{e,j}(t)) \varphi_j.\end{aligned}$$

Where $\phi_{i,j}$, $\phi_{e,j}$ and $w_j \in \mathbb{R}$ are the unknown expansion coefficients $\forall i, j = 1, \dots, N_h$. Then, we introduce the matrices:

$$\left. \begin{array}{lcl} (V_l)_{ij} &= \int_{\Omega} \nabla \varphi_j \cdot \Sigma_l \nabla \varphi_i, \\ (I_l^T)_{ij} &= \sum_{F \in F_h^I} \int_F \{\{\Sigma_l \nabla \varphi_j\}\} \cdot [[\varphi_i]], \\ (I_l)_{i,j} &= \sum_{F \in F_h^I} \int_F [[\varphi_j]] \cdot \{\{\Sigma_l \nabla \varphi_i\}\}, \\ (S_l)_{i,j} &= \sum_{F \in F_h^I} \int_F \Gamma_l [[\varphi_j]] \cdot [[\varphi_i]], \end{array} \right\} \quad \begin{array}{l} A_l = (V_l - I_l^T - \theta I_l + S_l) \\ l = i, e, \end{array}$$

$$\Gamma_l|_F = (n_F^T \Sigma_l n_F) \Gamma, \quad n_F \text{ outward normal vector of } F,$$

We also define:

$$A_i \in \mathbb{R}^{N_h \times N_h} \quad \text{Intra-cellular stiffness matrix},$$

$$A_e \in \mathbb{R}^{N_h \times N_h} \quad \text{Extra-cellular stiffness matrix},$$

$$M_{ij} = \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \varphi_j \varphi_i \quad \text{Mass matrix},$$

$$C(u_h)_{ij} = \sum_{\mathcal{K} \in \tau_h} \int_{\mathcal{K}} \chi_m k(u_h - 1)(u_h - a) \varphi_j \varphi_i \quad \text{Non-linear matrix},$$

$$\mathbf{F}_l = \begin{bmatrix} F_{i,l} \\ F_{e,l} \end{bmatrix} = \begin{bmatrix} \int_{\Omega} I_i^{ext} \varphi_l - \sum_{F \in F_h^B} \int_F b_i \varphi_l \\ - \int_{\Omega} I_e^{ext} \varphi_l - \sum_{F \in F_h^B} \int_F b_e \varphi_l \end{bmatrix}.$$

Therefore, our semi-discrete algebraic formulation is as follows:

Problem 4 (DG algebraic formulation). Find $\Phi_h(t) = [\phi_i^h(t), \phi_e^h(t)]^T \in [V_h^n]^2$ and $w_h(t) \in V_h^n$ for any $t \in (0; T]$ such that:

$$\begin{cases} \chi_m C m M \dot{V}_m^h + A_i \phi_i^h + C(V_m^h) V_m^h + \chi_m M w_h = F_i^h, \\ -\chi_m C m M \dot{V}_m^h + A_e \phi_e^h - C(V_m^h) V_m^h - \chi_m M w_h = F_e^h, \\ M \dot{w}_h(t) = \epsilon M (V_m^h(t) - \gamma w_h(t)), \end{cases}$$

Supplement with suitable initial conditions. An alternative and more compact version is given by:

Problem 5 (DG algebraic formulation - 2). Find $\Phi_h(t) = [\phi_i^h(t), \phi_e^h(t)]^T \in [V_h^n]^2$ and $w_h(t) \in V_h^n$ for any $t \in (0; T]$ such that:

$$\begin{cases} \chi_m C_m \begin{bmatrix} M & -M \\ -M & M \end{bmatrix} \begin{bmatrix} \dot{\phi}_i^h(t) \\ \dot{\phi}_e^h(t) \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & A_e \end{bmatrix} \begin{bmatrix} \phi_i^h(t) \\ \phi_e^h(t) \end{bmatrix} + \begin{bmatrix} C(V_m^h) & -C(V_m^h) \\ -C(V_m^h) & C(V_m^h) \end{bmatrix} \begin{bmatrix} \phi_i^h(t) \\ \phi_e^h(t) \end{bmatrix} + \\ + \chi_m \begin{bmatrix} M & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} w_h(t) \\ w_h(t) \end{bmatrix} = \begin{bmatrix} F_i^h \\ F_e^h \end{bmatrix}, \\ \dot{w}_h(t) = \epsilon (V_m^h(t) - \gamma w_h(t)). \end{cases}$$

3 Implementation details: the Dubiner Basis

3.1 Analytical aspects

So far, we have described a general semi-discrete discontinuous formulation without examining which basis to use to generate the V_h^n space. Usually, the common choice consists in the classical hat functions from FEMs. It is also one of the simplest choices, for this reason our provided code was initially implemented with this basis. However, the very novelty of this study is the adoption of a new kind of basis, completely different from the previous and commonly known as "*Dubiner Basis*"[7], which is well suited to high-order approximations.

How we will see soon, the peculiarity of this family of functions is that it consists of orthogonal polynomials defined on the reference triangle

$$\hat{K} = \{(\xi, \eta) : \xi, \eta \geq 0, \xi + \eta \leq 1\},$$

and not on the reference square

$$\hat{Q} = \{(a, b) : -1 \leq a \leq 1, -1 \leq b \leq 1\}.$$

Formally, if we consider the transformation from \hat{Q} to \hat{K} is given by:

$$\xi = \frac{(1+a)(1-b)}{4}, \eta = \frac{(1+b)}{2}, \quad (2)$$

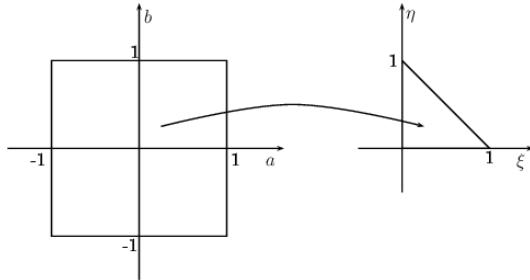


Figure 3: Transformation between the reference square (\hat{Q}) to the reference triangle (\hat{K})

the Dubiner basis is the transformation of a suitable basis initially defined on the reference square. This initial basis is simply obtained with a two dimensional modified tensor product of the Jacobi polynomials on the interval $(-1, 1)$, as described in the following definition.

Definition 1 (Jacobi polynomials). *The Jacobi polynomials of coefficients $\alpha, \beta \in \mathbb{R}$ evaluated in $z \in (-1, 1)$ are:*

$$- n = 0 \\ J_0^{\alpha, \beta}(z) = 1,$$

- $n = 1$

$$J_1^{\alpha,\beta}(z) = \frac{1}{2}(\alpha - \beta + (\alpha + \beta + 2) \cdot z),$$

- $n \geq 2$

$$\begin{aligned} J_n^{\alpha,\beta}(z) = & \sum_{k=2}^n \left[\frac{(2k+\alpha+\beta-1)(\alpha^2-\beta^2)}{2k(k+\alpha+\beta)(2k+\alpha+\beta-2)} + \right. \\ & \frac{(2k+\alpha+\beta-2)(2k+\alpha+\beta-1)(2k+\alpha\beta)}{2k(k+\alpha+\beta)(2k+\alpha+\beta-2)} J_{k-1}^{\alpha,\beta}(z) + \\ & \left. - \frac{2(k+\alpha-1)(k+\beta-1)(2k+\alpha+\beta)}{2k(k+\alpha+\beta)(2k+\alpha+\beta-2)} J_{k-2}^{\alpha,\beta}(z) \right]. \end{aligned}$$

The main property of these polynomials is the following:

Proposition 1. $\{J_i^{\alpha,\beta}, i = 0, 1, 2, \dots\}$ is orthogonal with respect to the Jacobi weight $w(x) = (1-x)^\alpha(1+x)^\beta$, i.e.:

$$\int_{-1}^1 (1-x)^\alpha(1+x)^\beta J_m^{\alpha,\beta}(x) J_q^{\alpha,\beta}(x) dx = \frac{2}{2m+1} \delta_{mq} \quad \forall i, j, m, q \geq 0.$$

We can now define explicitly the Dubiner basis.

Definition 2 (Dubiner Basis). The Dubiner basis that generates the space $\mathbb{P}^p(\hat{K})$ of polynomials of degree p over the reference triangle is the set of functions:

$$\phi_{ij} : \hat{K} \rightarrow \mathbb{R},$$

$$\begin{aligned} \phi_{ij}(\xi, \eta) := & c_{ij}(1-b)^j J_i^{0,0}(a) J_j^{2i+1,0}(b) = \\ & = c_{ij} 2^j (1-\eta)^j J_i^{0,0}\left(\frac{2\xi}{1-\eta} - 1\right) J_j^{2i+1,0}(2\eta - 1), \end{aligned}$$

for $i, j = 0, \dots, p$ and $i + j \leq p$, where

$$c_{ij} := \sqrt{\frac{2(2i+1)(i+j+1)}{4^i}}$$

and $J_i^{\alpha,\beta}(\cdot)$ is the i -th Jacobi polynomial defined in (1).

As we have anticipated the following result holds, cf. [13].

Proposition 2. The Dubiner basis is orthonormal in $L^2(\hat{K})$ $\forall p \geq 0$:

$$\int_{\hat{K}} \phi_{ij}(\xi, \eta) \phi_{mq}(\xi, \eta) d\xi d\eta = \delta_{im} \delta_{jq} \quad \forall i, j, m, q \geq 0.$$

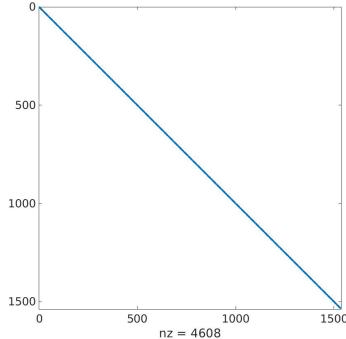


Figure 4: Non-zero elements in the mass matrix when adopting Dubiner basis

Notice that, thanks to the Proposition (2), the mass matrix of the DG space turns out to be diagonal. See Fig. (4) as an example.

It is noteworthy to point out that transformation (2) is bijective, it can be inverted but it needs some care. The natural inverse would be:

$$a = \frac{2\xi}{1-\eta} - 1 \quad b = 2\eta - 1,$$

that has already been used for the Definition (2). However, it is not defined for $\eta = 1$, that means for the sole point $(0,1)$ of the reference triangle. To solve this issue, it is enough to prolongate the function with continuity to this special point. For the code implementation, it is suggested avoiding evaluations in the exact point, which can be avoided using quadrature formula of Gaussian type.

In general, the orthogonality property implies some good numerical properties, not only the diagonalization of the mass matrix. For instance, in [3] interesting bounds for the conditional number have been proved. For this reason, we opted for this choice aiming to improve the previous results, at least from the space discretization side.

On the other hand, there are also some difficulties arising when one chooses to abandon the familiar FEM basis. First of all, the coefficients of a discretized function has only *modal* meaning and they no more represent the *nodal* values of the function itself. This fact needs some extra work when one needs to switch from the continuous functions to the discretized functions and viceversa, as it will be shown in the Section (3.2). Secondly, one can notice that these functions are not "boundary conditions friendly". What we mean is that, if compared to FEM basis, they have no particular properties on the boundary to let easily impose homogeneous boundary conditions. Thus, they should be again transformed, this time in a *boundary adapted* form. We address to [10] for a short description of this procedure. Fortunately, we do not need to set this transformation as in the discontinuous formulation Dirichlet boundary conditions are imposed only weakly. It means that the boundary conditions' choice does not imply the choice of

the vectorial space as in continuous Galerkin methods. The discretized space is always the same, only some terms in the weak formulation have in case of need to be changed. For this reason, the match of Discontinuous Galerkin and Dubiner Basis results to be particularly successful.

To conclude, we refer to [13] for the transformation and the definition of the Dubiner Basis with tetrahedra, i.e. in dimension $n = 3$.

3.2 Implementation

Our MATLAB code allows the user to select which basis to adopt (FEM or Dubiner) and the order of polynomials until $p=3$. We chose to call D_1, D_2, D_3 these 3 families of basis functions, thanks to the similarity to the P_1, P_2, P_3 finite element basis functions.

As explained in the Section (1.3), our starting point was the code implemented by the previous projects, namely the resolution of the Bidomain Model through FEM basis (P_1, P_2, P_3). As follows, our first goal was the implementation of methods to evaluate the Dubiner basis functions (D_1, D_2, D_3) and their gradients in the quadrature points. We omit the full code as it is not particular interesting: it barely follows the definitions of Section (3.1) with the addition of some technicalities. These scripts are: `eval_jacobi_polynomial.m`, `basis_legendre_dubiner.m`, `evalshape_tria_dubiner.m`.

Moreover, some conditional statements and some extra methods (as `matrix2D_dubiner.m`) were added to let the user easily switch from one basis to another (simply and once via `dati.m`).

More interesting are instead the scripts `dubiner_to_fem.m` and `fem_to_dubiner.m`, used to convert the Dubiner modal coefficients of the vector solution to the nodal values of the approximated function and viceversa. For this reason, they deserve some further explanations.

3.3 Switch from the modal expansion coefficients to the Lagrangian ones

One of the many advantages of the FEM basis is that there exists a bijection between the basis functions and some particular spacial points in such a way that the evaluation of a basis function in one of these points is equal to 1 only if that point is the one associated to the function and 0 otherwise, i.e.:

$$\psi_i(x_j) = \delta_{ij}. \quad (3)$$

Obviously, this property does not hold when we work with the Dubiner basis. Indeed, these functions are not associated to any mesh point. This implies that the Dubiner coefficients of a function $u \in V_h^p$ are not the evaluation at suitable points of the discretized function itself. They have a completely different meaning, they are now *modal* values

instead of being *nodal*. For this reason we introduced two new functions that transform the coefficients of the solution w.r.t. the FEM basis to the coefficients w.r.t. the Dubiner basis and viceversa.

Consider an element $\mathcal{K} \in \tau_h$ and $\{\psi_i\}_{i=1}^p, \{\phi_j\}_{j=1}^q$ as, respectively, the set of FEM functions and the set of Dubiner functions with support in \mathcal{K} . In addition, consider as $\{\hat{u}_i\}_{i=1}^p, \{\tilde{u}_j\}_{j=1}^q$ as, respectively, the FEM and Dubiner coefficients of the solution.

3.3.1 From Dubiner Basis to FEM Basis

Let us start from the transformation to the FEM coefficients. We now exploit the Property (3), i.e. the coefficient \hat{u}_i is nothing else but the evaluation of u_h on the i -th mesh point, then:

$$\hat{u}_i = \sum_{j=1}^q \tilde{u}_j \phi_j(x_i), \quad (4)$$

where x_i is the point associated to the ψ_i basis function.

This formula has been implemented in the lines from `dubiner_to_fem.m` script (Section (8.1)).

3.3.2 From FEM Basis to Dubiner Basis

Instead, to compute the coefficients conversely, we need to exploit the fact that the Dubiner Basis is L^2 -orthonormal (Proposition 2). We then need to compute a L^2 scalar product between the FEM discretized function and each Dubiner basis function. That is:

$$\tilde{u}_j = \int_{\mathcal{K}} u_h(x) \phi_j(x) dx = \int_{\mathcal{K}} \sum_{i=1}^p \hat{u}_i \psi_i(x) \phi_j(x) dx = \sum_{i=1}^p \left(\int_{\mathcal{K}} \psi_i(x) \phi_j(x) dx \right) \hat{u}_i, \quad j = 1, \dots \quad (5)$$

This slightly more difficult formula has been reproduced in `fem_to_dubiner.m` (Section (8.2)) using *Gauss-Legendre* quadrature formulas.

3.3.3 Final remarks

If the Dubiner functions are chosen as Galerkin basis, both the transformations are needed for the code implementation. Formula (4) is needed to plot and compute errors after the solution of the system. Formula (5) is instead needed to convert the FEM initial data u_0 into a vector of Dubiner coefficients before the resolution of the system.

In order to be rigorous, but also for the sake of simplicity, these transformations are implemented only from P_n to D_n , $n = 1, 2, 3$ and viceversa, where P_n means the FEM Basis of n-th polynomial degree, meanwhile D_n is for the Dubiner Basis of n-th polynomial degree. With this choice, the two bases generate the same space V_h^n and then the transformation infers only on the coefficients and not on the function's properties. Otherwise, decreasing n would mean to lose significant information, while increasing n does not substantially improve the quality of the solution as it initially belonged to a lower order space. Moreover, choosing the same degree for P and D implies several simplifications, for instance the same number of local nodes (nln). For this reason, both p and q are actually replaced with nln in the code.

4 Temporal discretization

So far, we have just studied the space discretization while a temporal discretization is still needed to totally discretize the Bidomain time-dependent problem. Thus, we divide the interval $(0, T]$ into N subintervals $(t^n, t^{n+1}]$ of length Δt such that $t^n = n\Delta t \quad \forall n = 0, \dots, N - 1$, we then assume our fully discretized solution as $V_m^n \approx V_m^h(t^n)$.

We have developed, implemented and tested 3 different temporal strategies that we will refer to as: *semi-implicit*, *Godunov operator-splitting* and *quasi-implicit operator-splitting*.

4.1 The Semi-implicit method

One of the most famous and used temporal scheme for a non-linear problem such as the Bidomain is certainly the *Semi-Implicit* scheme [11]. The basic idea is to treat most of the terms implicitly while treating the non-linear term semi-implicitly. Since the non-linear term is cubic, the best choice is to treat only one of these V_m terms implicitly, i.e.:

$$I_{ion}^{n+1} = k(V_m^n - a)(V_m^n - 1)V_m^{n+1} + w^{n+1},$$

at each time-step $n + 1$. Moreover, the gating variable ODE is treated implicitly with the exception of the term V_m :

$$M \frac{w^{n+1} - w^n}{\Delta t} = \epsilon M(V_m^n - \gamma w^{n+1}).$$

Therefore, we can transform the semi-discrete Problem (5) into:

$$\begin{cases} \chi_m C_m \begin{bmatrix} M & -M \\ -M & M \end{bmatrix} \begin{bmatrix} \phi_i^{n+1} - \phi_i^n \\ \phi_e^{n+1} - \phi_e^n \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & A_e \end{bmatrix} \begin{bmatrix} \phi_i^{n+1} \\ \phi_e^{n+1} \end{bmatrix} + \\ \begin{bmatrix} C(V_m^h) & -C(V_m^h) \\ -C(V_m^h) & C(V_m^h) \end{bmatrix} \begin{bmatrix} \phi_i^{n+1} \\ \phi_e^{n+1} \end{bmatrix} + \chi_m \begin{bmatrix} M & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} w^{n+1} \\ w^{n+1} \end{bmatrix} = \begin{bmatrix} F_i^{n+1} \\ F_e^{n+1} \end{bmatrix}, \\ M \frac{w^{n+1} - w^n}{\Delta t} = \epsilon M(V_m^n - \gamma w^{n+1}). \end{cases}$$

We remind that $V_m^n = \phi_i^n - \phi_e^n$. Separating known and unknown terms, we obtain:

$$\begin{cases} \left(\frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & -M \\ -M & M \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & A_e \end{bmatrix} + \begin{bmatrix} C(V_m^n) & -C(V_m^n) \\ -C(V_m^n) & C(V_m^n) \end{bmatrix} \right) \begin{bmatrix} \phi_i^{n+1} \\ \phi_e^{n+1} \end{bmatrix} = \\ \begin{bmatrix} F_i^{n+1} \\ F_e^{n+1} \end{bmatrix} - \chi_m \begin{bmatrix} M & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} w^{n+1} \\ w^{n+1} \end{bmatrix} + \frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} V_m^n \\ V_m^n \end{bmatrix}, \\ (\frac{1}{\Delta t} + \epsilon \gamma) M w^{n+1} = \epsilon M V_m^n + \frac{M}{\Delta t} w^n. \end{cases}$$

If we define:

- $B = \frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & -M \\ -M & M \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & A_e \end{bmatrix},$
- $C_{nl}(V_m^n) = \begin{bmatrix} C(V_m^n) & -C(V_m^n) \\ -C(V_m^n) & C(V_m^n) \end{bmatrix},$
- $r^{n+1} = \begin{bmatrix} F_i^{n+1} \\ F_e^{n+1} \end{bmatrix} - \chi_m \begin{bmatrix} M & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} w^{n+1} \\ w^{n+1} \end{bmatrix} + \frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & -M \\ -M & M \end{bmatrix} \begin{bmatrix} \phi_i^n \\ \phi_e^n \end{bmatrix}.$

we get the system in its final form.

Problem 6 (Semi-implicit discretized system). *Find $\Phi^{n+1} = [\phi_i^{n+1} \phi_e^{n+1}]^T$ and w^{n+1} $\forall n = 0, \dots, T-1$ such that:*

$$\begin{cases} (\frac{1}{\Delta t} + \epsilon\gamma) M w^{n+1} = \epsilon M V_m^n + \frac{M}{\Delta t} w^n, \\ (B + C_{nl}(\Phi^n)) \Phi^{n+1} = r^{n+1}. \end{cases}$$

The implementation can be found at the Section (8.3).

4.2 The Godunov operator-splitting

The main feature of a general operator-splitting method is the sub-division of the problem into two different problems to be solved sequentially. This is possible and justified when the original functional operator L is splitted into 2 different operators such that $L(u) = L_1(u) + L_2(u)$. Two operator-splitting methods have been implemented, the first is of *Godunov* type and a detailed study together with its properties can be found in [14]. The formulation is:

Find $\hat{V}_m^{n+1}, \phi_i^{n+1}, \phi_e^{n+1}, w^{n+1}$ such that:

$$\begin{cases} \chi_m C_m M \frac{\hat{V}_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^n + \chi_m M w^n = 0, \\ \frac{w^{n+1} - w^n}{\Delta t} = \epsilon(V_m^n - \gamma w^n). \end{cases}$$

$$\begin{cases} \chi_m C_m M \frac{V_m^{n+1} - \hat{V}_m^{n+1}}{\Delta t} + A_i \phi_i^{n+1} = F_i^{n+1}, \\ -\chi_m C_m M \frac{V_m^{n+1} - \hat{V}_m^{n+1}}{\Delta t} + A_e \phi_e^{n+1} = F_e^{n+1}. \end{cases}$$

Putting in a unique system:

$$\begin{cases} \chi_m C_m M \frac{V_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^n + \chi_m M w^n + A_i \phi_i^{n+1} = F_i^{n+1}, \\ \chi_m C_m M \frac{V_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^n + \chi_m M w^n - A_e \phi_e^{n+1} = -F_e^{n+1}, \\ w^{n+1} = (1 - \epsilon \gamma \Delta t) w^n + \epsilon \Delta t V_m^n. \end{cases} \quad (6)$$

The equations in the system (6) can be rewritten as:

$$\begin{cases} \left(\frac{\chi_m C_m}{\Delta t} M + A_i \right) \phi_i^{n+1} - \frac{\chi_m C_m}{\Delta t} M \phi_e^{n+1} = F_i^{n+1} - \chi_m M w^n + \left(\frac{\chi_m C_m}{\Delta t} M - C(V_m^n) \right) V_m^n, \\ \frac{\chi_m C_m}{\Delta t} M \phi_i^{n+1} - \left(\frac{\chi_m C_m}{\Delta t} M + A_e \right) \phi_e^{n+1} = -F_e^{n+1} - \chi_m M w^n + \left(\frac{\chi_m C_m}{\Delta t} M - C(V_m^n) \right) V_m^n, \\ w^{n+1} = (1 - \epsilon \gamma \Delta t) w^n + \epsilon \Delta t V_m^n. \end{cases}$$

Then, we obtain the final form:

Problem 7 (Godunov operator-splitting discretized system). *Find $\Phi^{n+1} = [\phi_i^{n+1} \phi_e^{n+1}]^T$ and $w^{n+1} \quad \forall n = 0, \dots, N-1$ such that:*

$$\begin{cases} \left(\frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & -M \\ M & -M \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & -A_e \end{bmatrix} \right) \begin{bmatrix} \phi_i^{n+1} \\ \phi_e^{n+1} \end{bmatrix} = \begin{bmatrix} F_i^{n+1} \\ -F_e^{n+1} \end{bmatrix} + \\ -\chi_m \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} w^n \\ w^n \end{bmatrix} + \left(\frac{\chi_m C_m}{\Delta t} \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix} - \begin{bmatrix} C(V_m^n) & 0 \\ 0 & C(V_m^n) \end{bmatrix} \right) \begin{bmatrix} V_m^n \\ V_m^n \end{bmatrix}, \\ w^{n+1} = (1 - \epsilon \gamma \Delta t) w^n + \epsilon \Delta t V_m^n. \end{cases}$$

The implementation is written at the Section (8.4).

4.3 The Quasi-implicit operator-splitting

The aim of a quasi-implicit operator splitting is to treat implicitly all the terms except the cubic one. Even if it cannot be defined as a fully implicit method, we hope to achieve more stability if compared to the previous Godunov-kind scheme. This time, the formulation turns to be:

Find $\tilde{V}_m^{n+1}, \phi_i^{n+1}, \phi_e^{n+1}, w^{n+1}$ such that:

$$\begin{cases} \chi_m C_m M \frac{\tilde{V}_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^{n+1} + \chi_m M w^{n+1} = 0, \\ \frac{w^{n+1} - w^n}{\Delta t} = \epsilon (V_m^{n+1} - \gamma w^{n+1}). \end{cases}$$

$$\begin{cases} \chi_m C_m M \frac{V_m^{n+1} - \tilde{V}_m^{n+1}}{\Delta t} + A_i \phi_i^{n+1} = F_i^{n+1}, \\ -\chi_m C_m M \frac{V_m^{n+1} - \tilde{V}_m^{n+1}}{\Delta t} + A_e \phi_e^{n+1} = F_e^{n+1}. \end{cases}$$

Putting into a unique system, we obtain:

$$\begin{cases} \chi_m C_m M \frac{V_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^{n+1} + \chi_m M w^{n+1} + A_i \phi_i^{n+1} = F_i^{n+1}, \\ \chi_m C_m M \frac{V_m^{n+1} - V_m^n}{\Delta t} + C(V_m^n) V_m^{n+1} + \chi_m M w^{n+1} - A_e \phi_e^{n+1} = -F_e^{n+1}, \\ \frac{w^{n+1} - w^n}{\Delta t} = \epsilon (V_m^{n+1} - \gamma w^{n+1}). \end{cases} \quad (7)$$

If we define:

- $Q_k = \frac{\chi_m C_m}{\Delta t} M + C(V_m^n) + \frac{\epsilon \chi_m \Delta t}{1 + \epsilon \gamma \Delta t} M$,
- $R_k = \frac{\chi_m C_m}{\Delta t} M V_m^n - \frac{\chi_m}{1 + \epsilon \gamma \Delta t} M w^n$.

the equations in system (7) can be written as:

$$\begin{aligned} & \chi_m C_m M \frac{\phi_i^{n+1} - \phi_e^{n+1} - V_m^n}{\Delta t} + C(V_m^n) (\phi_i^{n+1} - \phi_e^{n+1}) + \\ & + \chi_m M \left(\frac{w^n + \epsilon \Delta t (\phi_i^{n+1} - \phi_e^{n+1})}{1 + \epsilon \gamma \Delta t} \right) + A_i \phi_i^{n+1} = F_i^{n+1}, \end{aligned} \quad (8)$$

$$\Rightarrow (Q_k + A_i) \phi_i^{n+1} - Q_k \phi_e^{n+1} = R_k + F_i^{n+1},$$

$$\begin{aligned} & \chi_m C_m M \frac{\phi_i^{n+1} - \phi_e^{n+1} - V_m^n}{\Delta t} + C(V_m^n) (\phi_i^{n+1} - \phi_e^{n+1}) + \\ & + \chi_m M \left(\frac{w^n + \epsilon \Delta t (\phi_i^{n+1} - \phi_e^{n+1})}{1 + \epsilon \gamma \Delta t} \right) - A_e \phi_e^{n+1} = -F_e^{n+1}, \end{aligned} \quad (9)$$

$$\Rightarrow Q_k \phi_i^{n+1} - (Q_k + A_e) \phi_e^{n+1} = R_k - F_e^{n+1},$$

$$w^{n+1} = \frac{w^n + \epsilon \Delta t (\phi_i^{n+1} - \phi_e^{n+1})}{1 + \epsilon \gamma \Delta t}. \quad (10)$$

The final system becomes:

Problem 8 (Quasi-implicit operator-splitting discretized system). Find $\Phi^{n+1} = [\phi_i^{n+1} \phi_e^{n+1}]^T$ and $w^{n+1} \quad \forall n = 0, \dots, N-1$ such that:

$$\begin{cases} \left(\begin{bmatrix} Q_k & -Q_k \\ Q_k & -Q_k \end{bmatrix} + \begin{bmatrix} A_i & 0 \\ 0 & -A_e \end{bmatrix} \right) \begin{bmatrix} \phi_i^{n+1} \\ \phi_e^{n+1} \end{bmatrix} = \begin{bmatrix} R_k \\ R_k \end{bmatrix} + \begin{bmatrix} F_i^{n+1} \\ -F_e^{n+1} \end{bmatrix}, \\ w^{n+1} = \frac{w^n + \epsilon \Delta t (\phi_i^{n+1} - \phi_e^{n+1})}{1 + \epsilon \gamma \Delta t}. \end{cases}$$

The implementation is at the Section (8.5).

5 About uniqueness of the potentials

5.1 Analytical concepts

Observing for a moment the Bidomain problem analytical formulation (Problem (2)), we can immediately realize that the intracellular and the extracellular potentials appear only through their difference V_m or their gradient. This means that there cannot be uniqueness for the two functions. Namely:

$$\begin{aligned} \phi_i, \phi_e \text{ classical solutions of Bidomain} \Rightarrow \phi_i + \varphi, \phi_e + \varphi \text{ are solutions as well} \\ \forall \varphi : [0, T] \rightarrow \mathbb{R} \quad \text{sufficiently regular.} \end{aligned} \quad (11)$$

However, this fact should not surprise nor confuse the reader. First of all, we remind again that in [5] and [6] there are proofs for the V_m and w uniqueness, then this is taken for granted. Secondly, this statement reflects the physical intuition of the problem: cellular dynamics is not involved by potentials exact values but instead from their difference, in addition a potential value is nonsense if a convention value to compare it with has not been set. The dependence on time can be interpreted as follows: if, at any time instant, we change the conventional potential value, the dynamics of the problem does not change.

Moreover, we can give this simple result to show that the solutions of the form of equation (11) are also the only admissible:

Theorem 1. *For the Bidomain problem coupled with Fitzhugh-Nagumo model with Neumann boundary conditions (Problem (2)), the solutions ϕ_i, ϕ_e are unique up to a constant depending only on time.*

Proof. We remind that existence and uniqueness for V_m and w have already been proved in [5]. Suppose now there exist two couples $(\phi_i^1, \phi_e^1), (\phi_i^2, \phi_e^2)$ of potentials solutions of the Bidomain problem. If V_m is unique, then there must exist a unique value of V_m such that:

$$\phi_i^1 - \phi_e^1 = \phi_i^2 - \phi_e^2 = V_m,$$

Then, we define a function $\varphi : \Omega \times [0, T] \rightarrow \mathbb{R}$ as:

$$\varphi := \phi_i^1 - \phi_i^2 = \phi_e^1 - \phi_e^2,$$

If we consider the Problem (2), the following equations must hold:

$$\begin{cases} \chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_i \nabla \phi_i^1) + \chi_m I_{ion}(V_m, w) = I_i^{ext}, & \text{in } \Omega_{mus} \times (0, T], \\ \chi_m C_m \frac{\partial V_m}{\partial t} - \nabla \cdot (\Sigma_i \nabla \phi_i^2) + \chi_m I_{ion}(V_m, w) = I_i^{ext}, & \text{in } \Omega_{mus} \times (0, T], \\ \Sigma_i \nabla \phi_i^1 \cdot n = b_i, & \text{on } \partial \Omega_{mus} \times (0, T], \\ \Sigma_i \nabla \phi_i^2 \cdot n = b_i, & \text{on } \partial \Omega_{mus} \times (0, T]. \end{cases}$$

Subtracting the first two equations and the second two we obtain:

$$\begin{cases} -\nabla \cdot (\Sigma_i \nabla \varphi) = 0, & \text{in } \Omega_{mus} \times (0, T], \\ \Sigma_i \nabla \varphi \cdot n = 0, & \text{on } \partial \Omega_{mus} \times (0, T]. \end{cases}$$

That is a classical *Laplace problem* with homogeneous Neumann boundary conditions. From [12], we know that the solution set is composed by all constant terms (remember that Σ_i is positive). However, we must pay attention to the fact that φ is a time-dependent function, even if time does not compare in the system. Thus, we can state:

$$\exists \tilde{\varphi} : [0, T] \rightarrow \mathbb{R} \text{ such that } \varphi(x, t) = \tilde{\varphi}(t) \quad \forall x \in \Omega, \forall t \in [0, T].$$

To conclude, we can observe now that if these two couples of solutions exist, then:

$$\phi_i^1 - \phi_i^2 = \phi_e^1 - \phi_e^2 = \tilde{\varphi} \quad \forall x \in \Omega, \forall t \in [0, T].$$

□

Remark. *For what concerns the regularity of φ , we can certainly state that, as a difference of two sufficiently regular functions, it belongs to the same class of regularity of the potentials if restricted to the sole time variable.*

We can conclude this analytical digression with an accomplished necessary and sufficient condition for the potentials solutions.

Corollary 1. *Suppose the couple (ϕ_i, ϕ_e) is a classical solution of the problem of Problem (2) (for a certain w). The couple $(\tilde{\phi}_i, \tilde{\phi}_e)$ of sufficiently regular real functions defined in $\Omega \times [0, T]$ is another couple solution if and only if both $\tilde{\phi}_i, \tilde{\phi}_e$ differ respectively from ϕ_i, ϕ_e for a time-dependent function φ that belongs to the union of the regularity classes of the previous functions if restricted to time variable.*

Proof. The regularity statement is trivial and already discussed. The right implication is due to the previous theorem. Finally, the left implication follows what has been shown in equation (11): it is enough to insert $\phi_i + \varphi$ and $\phi_e + \varphi$ in the Bidomain system to find out that φ disappears and the remaining system is the same as the one with ϕ_i, ϕ_e , thus solved by hypothesis. □

5.2 Numerical correction

Previous analytical results are crucial for what concerns the numerical computations since the Bidomain problem turns now to be *not exactly* well-posed. Even if, in general, the right V_m is most of the times achieved thanks to its uniqueness, our aim is to impose a further condition on the ϕ unknowns for the following two reasons:

1. To pursue the achievement of the exact ϕ_i, ϕ_e and not their values minus a constant.
Useful for instance for the error analysis.
2. To strengthen our Galerkin formulation that currently derives from a ill-posed problem and, for this reason, may show bad features as the generation of a ill-conditioned system or even a non-solvable system.

We firstly observe that the additional condition should be applied only to one of the potentials, for instance to ϕ_i . Indeed, the difference of the two possible solutions is φ for both intracellular and extracellular potentials. Imposing φ at each time-step implies the uniqueness imposition for both ϕ_i, ϕ_e .

The most common and simple strategies are the following:

1. Imposition of the value of the function in a specific point.

$$\phi_i(\bar{x}, t) = \varphi(t) \quad \forall t \in [0, T].$$

2. Imposition of the function mean value.

$$\int_{\Omega} \phi_i(x, t) dx = \varphi(t) \quad \forall t \in [0, T].$$

Notice that the first strategy would be useless if we keep working with an analytical and abstract weak formulation in Sobolev spaces. However, in the numerical context, we can assume certain regularities for the solution that let it makes sense and be the most common choice for numerical implementations.

As we will examine later on, it is demanding to implement the first strategy in a Dubiner context without losing the main system properties. Thus, we slightly change the first strategy and we instead opt for the imposition of a vector solution coefficient. For what concerns the FEM basis, this has the same meaning as before, provided that \bar{x} is not a whatever point but a *dof* point. On the other hand, for Dubiner basis, this has a completely different and abstract meaning: we remind that this time it has the role of modal coefficient.

Consider $\{u_j\}_{j=1\dots N_h}$ as the list of the vector solution. As a consequence, we give the numerical version of the previous strategies:

1. Imposition of a coefficient of the vector solution.

$$u_l^n = \varphi(t^n) \quad \forall n \in \{1, N\}.$$

2. Imposition of the function mean value.

$$\sum_{j=1\dots N_h} u_j^n w_j = \varphi(t^n) \quad \forall n \in \{1, N\}.$$

where l is a fixed value $\in \{1, N_h\}$ and w_j stands for a suitable weight (that depends on the mesh geometry, the basis choice and the quadrature formula chosen).

We remind that our aim is to impose such conditions *directly* into the system. The easiest way would certainly be to impose these conditions *after* the system resolution, as it has been reproduced in other past works. However, in this case, some issues related to the ill-posedness arise, especially ill-conditioning. Then, in the next sections we will illustrate how we managed to impose potential uniqueness only changing matrices and vectors coefficients before the resolution.

5.2.1 Implementation of the first coefficient imposition

For simplicity, we choose $l = 1$. Then, u_l^n has the meaning of:

- Value of u in the first dof point, \forall timestep n (FEM)
- Fourier coefficient of u w.r.t. the first Dubiner basis function, \forall timestep n (Dubiner)

What follows will be independent from basis choice. Suppose $c \in \mathbb{R}$ is the value to impose in the system $Au = \vec{b}$ for a certain timestep n . Since first coefficient occupies the first cell in the unknown vector and influences other coefficient values only through the first matrix column, we can switch the system from:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ a_{31} & a_{32} & \dots & a_{3N} \\ \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ b_N \end{bmatrix}$$

to:

$$\tilde{A} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & a_{22} & \dots & a_{2N} \\ 0 & a_{32} & \dots & a_{3N} \\ \dots & \dots & \dots & \dots \\ 0 & a_{N2} & \dots & a_{NN} \end{bmatrix} \quad \tilde{b} = \begin{bmatrix} c \\ b_2 - a_{21}c \\ b_3 - a_{31}c \\ \dots \\ b_N - a_{N1}c \end{bmatrix}$$

This is certainly correct since in the first system line $u_1 = c$ is automatically imposed and, in the other lines, u_1 is no more treated as unknown but as a known data and then moved to the r.h.s. of the system.

The very advantage of this procedure is the conservation of A symmetry. As we have previously anticipated, we discarded the nodal value strategy because, using Dubiner basis, we would have lost this crucial property.

Moreover, the value c can be freely chosen, for instance from the exact solution (when error analysis needs to be executed) or a conventional fixed value as zero.

On the other hand, there are two main disadvantages. First of all, the system first line information has been deleted during this procedure. However, if the mesh is composed by many elements, this information is not essential and the solution behavior is practically the same as this information were provided.

Secondly, if the initial system is hugely ill-conditioned or even non-solvable (determinant could approximate the machine epsilon when we have homogeneous boundary conditions and/or no forcing terms), this imposition may have an overshooting effect that unbalances the solution. For these problems, a global imposition has to be adopted and that is the reason why we implemented the more complicated mean value imposition strategy.

The coefficient imposition procedure has been implemented in the script `assign_phi_i.m` (Section (8.6)) that takes the c value from the exact solution.

To conclude this section, it is interesting to observe that the numerical imposition is done at *every* time-step. This is confirmed from previous analytical theory as the difference φ is a constant but depending on time: therefore, it is needed to fix it at every time-step.

5.2.2 An analytical motivation for the mean value imposition method

It is easy to realize that the procedure in Section (5.2.1) cannot be replicated for the mean unless losing symmetry. For instance, in the case of FEM basis and zero mean, a first line full of ones would imply also a first column full of ones and thus the resolution would be compromised. We should look for a different strategy. Let us start with a simple reference problem:

Problem 9 (Reference zero-mean problem - strong form). *Let Ω be an open, bounded and sufficiently regular domain, $f \in C^0(\bar{\Omega})$. Find $u \in C^2(\Omega) \cap C^1(\bar{\Omega})$ such that:*

$$\begin{cases} -\Delta u = f, & \text{in } \Omega, \\ \int_{\Omega} u = 0, & \text{in } \Omega, \\ \nabla u \cdot n = 0, & \text{on } \partial\Omega. \end{cases}$$

For our scopes, it is convenient to move to the variational formulation:

Problem 10 (Reference zero-mean problem - weak form). *Let Ω be an open and bounded set, $f \in L^2(\Omega)$. Find $u \in H^1(\Omega)$ such that:*

$$\begin{cases} \int_{\Omega} \nabla u \cdot \nabla v = \int_{\Omega} fv, & \forall v \in H^1(\Omega), \\ \int_{\Omega} u = 0. \end{cases}$$

As usual, the regularity of f, Ω imply that the weak solution is the classical solution as well. For this reason, let us focus only on the weak form. In addition, observe that if Ω is bounded, then $H^1(\Omega) \subset L^2(\Omega) \subset L^1(\Omega)$, therefore the second equation is justified. The next step is the study of the well-posedness.

Lemma 1. *The Problem (10) admits a weak solution u if and only if the compatibility condition $\int_{\Omega} f = 0$ holds, in other words f is a zero-mean function. Moreover, if u exists, it is unique and it minimizes the Laplace energy functional $J(u) = \frac{1}{2} \int_{\Omega} |\nabla u|^2 - \int_{\Omega} fu$*

Proof. Consider for the moment the first equation only. It is the Laplace problem with homogeneous Neumann boundary conditions. In a more general form, it is equivalent to a specific reaction-diffusion problem:

$$\begin{cases} -\Delta u + \alpha u = f, & \text{in } \Omega, \\ \nabla u \cdot n = 0, & \text{on } \partial\Omega, \end{cases}$$

with $\alpha = 0$.

We have already discussed that $\alpha = 0$ is an eigenvalue of the Laplace operator with Neumann boundary conditions and its eigenspace is composed by all and only constant terms. From theorem 7.1.14 in [8], we can state that, since α belongs to the spectrum and $f \in L^2(\Omega)$, existence of the weak solution holds if and only if the compatibility condition:

$$\int_{\Omega} f = \int_{\partial\Omega} g = \int_{\partial\Omega} 0 = 0$$

holds. Then we solved the point about existence of the weak solution.

For what concerns the uniqueness, we know from the same theorem that u is unique except for other functions that differ from u for an eigenfunction associated to $\alpha = 0$. Since the eigenfunctions of zero are the functions that are constant *a.e.*, we can state that the weak solution is unique minus a constant term. Then, if we add the second equation $\int_{\Omega} u = 0$, we achieve existence and uniqueness of the solution.

Suppose now u is the weak solution and v another function $\in H^1(\Omega)$. Then:

$$\begin{aligned} & \exists w \in H^1(\Omega) : v = w + u \\ \Rightarrow J(v) &= J(u + w) = \frac{1}{2} \int_{\Omega} |\nabla u + \nabla w|^2 - \int_{\Omega} fu - \int_{\Omega} fw = \\ &= \underbrace{\frac{1}{2} \int_{\Omega} |\nabla u|^2 - \int_{\Omega} fu}_{J(u)} + \underbrace{\frac{1}{2} \int_{\Omega} |\nabla w|^2 + \int_{\Omega} \nabla u \cdot \nabla w - \int_{\Omega} fw}_{=0, \text{ by def of weak solution}} = \\ &= J(u) + \frac{1}{2} \int_{\Omega} |\nabla w|^2 \geq J(u). \end{aligned}$$

□

Remark. *The minimization of the functional J for the Laplace problem is a known fact. However, in this case where the sole Laplace-Neumann problem is not well-posed, this result was not trivial and thus it needed a check. Indeed, it is noteworthy to underline that minimization property holds but in a slightly different way: u is not the absolute minimum point, every $u + \xi, \xi \in \mathbb{R}$ reaches the same minimum.*

Previous well-posedness and minimization results implies that, if u solves Problem (10), then it is unique and, moreover, it is the unique zero mean value function that minimizes the functional $J(u)$. Thus, we can transform Problem (10) in another formulation:

Problem 11 (Reference mean-value problem - 2). *Find $u \in H^1(\Omega)$ such that*

$$\begin{cases} J(u) = \min_{v \in H^1(\Omega)} J(v), \\ I(u) = 0. \end{cases}$$

where $f \in L^2(\Omega)$ and

- $J(u) = \frac{1}{2} \int_{\Omega} |\nabla u|^2 - \int_{\Omega} f u,$
- $I(u) = \int_{\Omega} u.$

Surely a solution of Problem (10) is a solution of Problem (11). The converse implication could be easily proved too. Then, the two problems are well-posed and completely equivalent. The advantage of the second form is that it consists in a minimization problem with constraints, a kind of problem that can be solved with generalized *Lagrange Multipliers*. It means that:

$$\exists \lambda \in \mathbb{R} \text{ such that } \langle J'(u), v \rangle + \lambda \langle I'(u), v \rangle = 0 \quad \forall v \in H^1(\Omega),$$

where J', I' are the *Frechét derivatives* of the two operators J, I and $\langle \cdot, \cdot \rangle$ represents the H^1 duality. Computing the derivatives, we indeed obtain:

$$\exists \lambda \in \mathbb{R} \text{ such that } \int_{\Omega} \nabla u \cdot \nabla v + \lambda \int_{\Omega} v = \int_{\Omega} f v \quad \forall v \in H^1(\Omega),$$

We can then formulate a third and last version of the reference problem:

Problem 12 (Reference mean-value problem - 3). *Find $u \in H^1(\Omega), \lambda \in \mathbb{R}$ such that:*

$$\begin{cases} \int_{\Omega} \nabla u \cdot \nabla v + \lambda \int_{\Omega} v = \int_{\Omega} f v, & \forall v \in H^1(\Omega), \\ \int_{\Omega} u = 0. \end{cases}$$

It may seem a very trivial result but actually it will be the very essence of our mean-value imposition strategy. First of all, let us check that existence and uniqueness properties have been conserved.

Lemma 2. Suppose that the assumptions on data of the Problem (10) are satisfied. Then there exists a couple solution (u, λ) to Problem (12). Moreover, $\lambda = 0$ and u is unique.

Proof. For what concerns existence, we can immediately realize that the solution u of Problem (10) solves the Problem (12) with $\lambda = 0$. Then the existence property holds because existence of Problem (10) has already been proved.

Suppose now there exist two couples $(u_1, 0), (u_2, \lambda)$ solutions of the problem and define $\varphi = u_2 - u_1$. Then:

$$\begin{cases} \int_{\Omega} \nabla u_1 \cdot \nabla v = \int_{\Omega} f v, & \forall v \in H^1(\Omega), \\ \int_{\Omega} \nabla u_2 \cdot \nabla v + \lambda \int_{\Omega} v = \int_{\Omega} f v, & \forall v \in H^1(\Omega), \\ \int_{\Omega} u_1 = \int_{\Omega} u_2 = 0. \end{cases}$$

Subtracting, we obtain:

$$\begin{cases} \int_{\Omega} \nabla \varphi \cdot \nabla v + \lambda \int_{\Omega} v = 0, & \forall v \in H^1(\Omega), \\ \int_{\Omega} \varphi = 0. \end{cases}$$

If we assign $v = \varphi \in H^1(\Omega)$, then:

$$\begin{cases} \int_{\Omega} |\nabla \varphi|^2 + \lambda \int_{\Omega} \varphi = \int_{\Omega} |\nabla \varphi|^2 = 0, & \forall v \in H^1(\Omega), \\ \int_{\Omega} \varphi = 0. \end{cases}$$

Then, $\|\nabla \varphi\|_{L^2} = 0$ implies φ constant. But, since it has zero mean, $\varphi = 0$ a.e. To conclude, if $u_1 = u_2$ a.e. as just proved, the previous system becomes:

$$\lambda \int_{\Omega} v = 0 \quad \forall v \in H^1(\Omega),$$

that trivially implies $\lambda = 0$.

□

This analytical digression was intended as a clarification of how Problem (12) can be considered as equivalent to the original Problem (10) (indeed, they are both well-posed and have the same solution). For this reason, the system modifications of the next section will be in some way justified by the previous results even if Bidomain problem is hugely more complicated than the simple Laplace problem.

5.2.3 Implementation of the mean value imposition

Following the Problem (12) new formulation, the basic idea is to consider λ as a new coefficient of the vector solution, for instance the last one. The vector u is now of dimension $N_h + 1$. Let us define $d_i = \int_{\Omega} \varphi_i$ where φ_i is the i -th basis function (both for FEM and Dubiner). Moreover, define c as the imposed value for the mean. Then the discretized problem at a certain time-step turns to be:

Problem 13 (Discretized mean-value imposition problem). *Find $\{u_i\}_{i=1\dots N_h+1}$ such that:*

$$\begin{cases} \sum_{i=1}^{N_h} u_i \int_{\Omega} \nabla \varphi_i \cdot \nabla \varphi_j + \lambda d_j = \int_{\Omega} f \phi_j, & \forall j = 1 \dots N_h, \\ \sum_{i=1}^{N_h} u_i d_i = c. \end{cases}$$

Reminding that $\lambda = u_{N_h+1}$, the previous problem consists in the system transformation from:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_N \end{bmatrix}$$

to:

$$\tilde{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} & d_1 \\ a_{21} & a_{22} & \dots & a_{2N} & d_2 \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NN} & d_N \\ d_1 & d_2 & \dots & d_N & 0 \end{bmatrix} \quad \tilde{b} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_N \\ c \end{bmatrix}$$

First of all, observe that symmetry is conserved. Moreover, this time no line has been deleted, all the information is conserved. To conclude, we remind from Lemma (2) that λ is an auxiliary unknown, its value is always zero.

The implementation of such transformation is carried out in the `assign_null_average.m` script (at Section (8.7)).

Some comments:

- `Nh = length(b)/2` at line 3 because the original system is a block matrix system. We remind that this transformation concerns only ϕ_i , then it applies to the first half of the system only.
- c is chosen to be zero (line 42), its value does not come from an exact solution because the mean-value strategy has been adopted only for realistic simulation with no exact solutions (see 5.2.4).

- We avoided to compute all d_i for FEM basis as they all have the same value.
- On the other hand, for Dubiner basis, d_i values are different. However, it is not needed to compute these values for all the global polynomials as they repeat for every element. For this reason, we only iterate over the local degrees of freedom.

5.2.4 Final remarks

As already discussed, the mean-value imposition was implemented and adopted only for very ill-conditioned systems. For all other cases, the coefficient imposition worked perfectly. This is why, in our research, we chose to adopt:

- the coefficient imposition for error analysis simulations (as boundary conditions and forcing terms were never homogeneous)
- the mean value imposition for realistic simulations (as boundary conditions and forcing terms were essentially homogeneous)

6 Error analysis results

6.1 Data

Starting from some example problems, we provide now an experimental error analysis that can show the efficacy and goodness of our numerical schemes. For all the simulations, the values to assign to constant parameters are the same proposed in [4] and used in [2] and [9]. Namely:

Table 1: Parameters for pseudo-realistic simulations

χ_m	10^5
Σ_i	$\begin{bmatrix} 0.12 & 0 \\ 0 & 0.12 \end{bmatrix}$
Σ_e	$\begin{bmatrix} 0.12 & 0 \\ 0 & 0.12 \end{bmatrix}$
C_m	10^{-2}
k	19.5
ε	1.2
γ	0.1
a	$13 \cdot 10^{-3}$

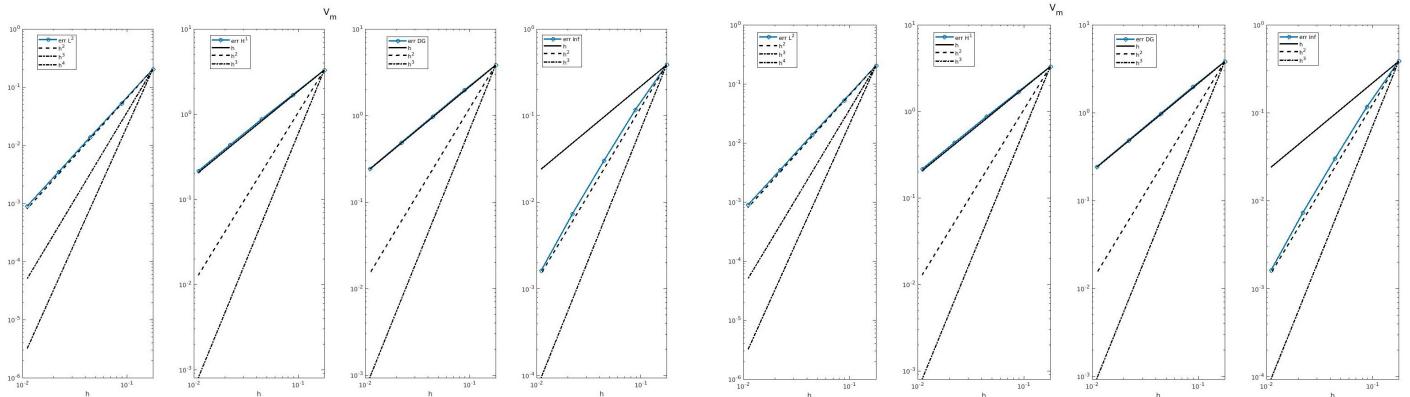
We initially consider as exact solutions $V_m(x, y, t) = \sin(2\pi x)\sin(2\pi y)e^{-5t}$ and $w(x, y, t) = \frac{\varepsilon}{\varepsilon\gamma-5}\sin(2\pi x)\sin(2\pi y)e^{-5t}$, in a domain $\Omega = [0, 1; 0, 1]$ and time interval $(0, 0.001]$. From these assumptions, we consequently obtain the suitable boundary conditions, initial conditions and forcing terms.

6.2 Comparison between Dubiner and FEM

At first, an error analysis related to the basis choice is shown. More precisely, we fix a polynomial order and we compare the errors of the Dubiner and FEM basis choosing this order for both. We expect to see similar results for each couple of basis. On the other hand, we expect to see different error orders for different polynomials orders.

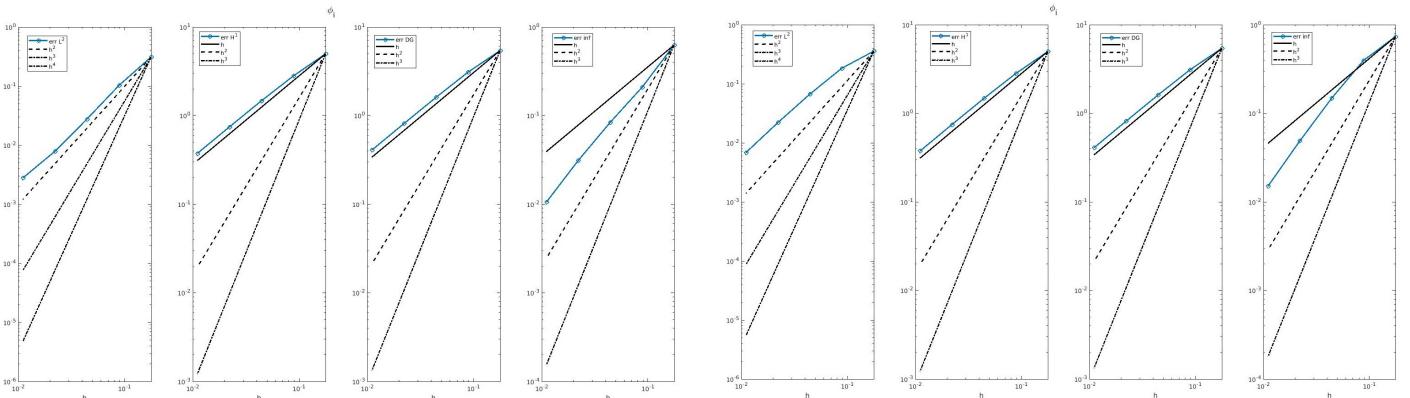
6.2.1 P1-D1

If 1 is chosen as polynomial order, the errors plots are the following. It is clear that the convergence is very alike since the errors between the two kinds of basis have the same trend. Then, in this case, results are exactly as expected and show a first order for H^1 and DG errors while second order for L^2 and L^∞ errors.



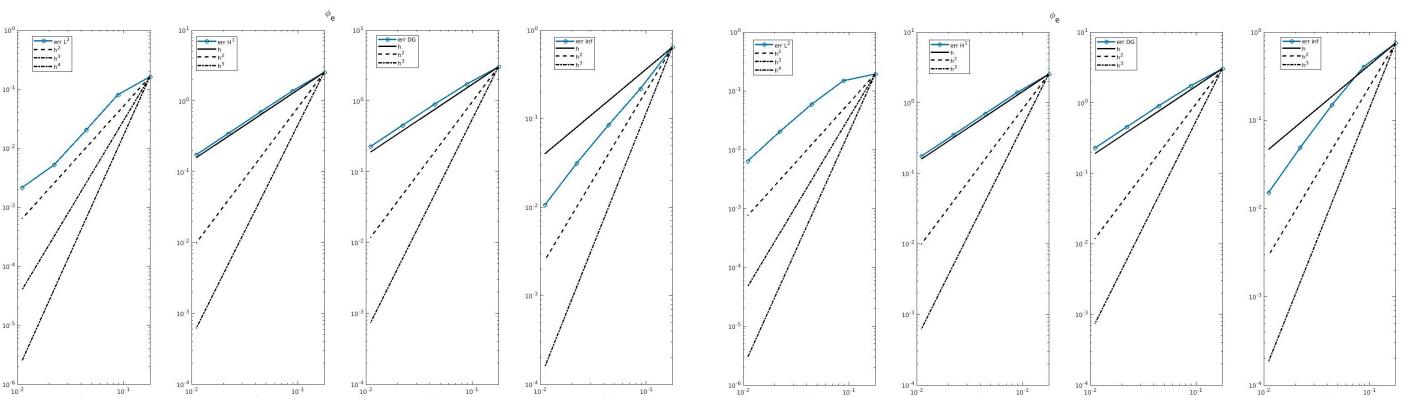
(a) Difference of potential (V_m) with D1

(b) Difference of potential (V_m) with P1



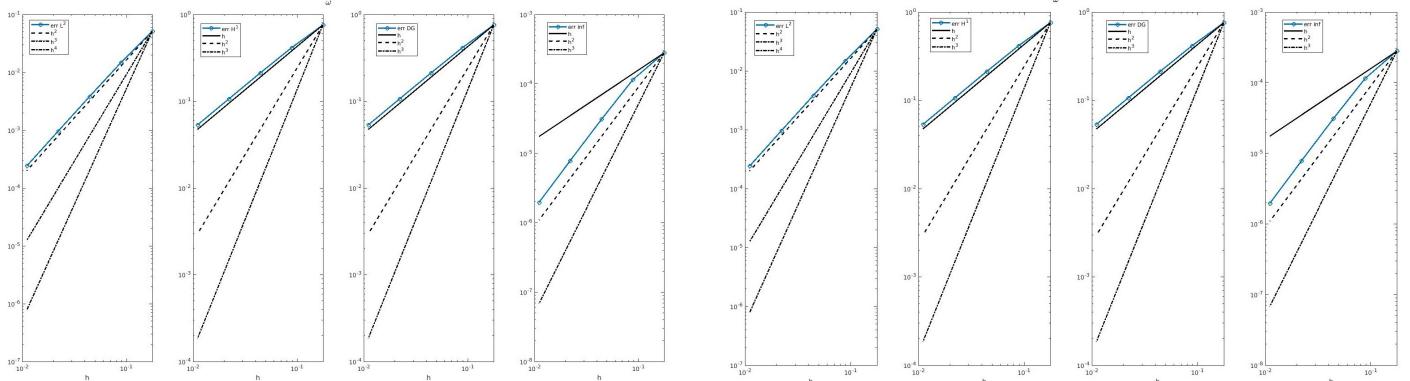
(a) Intracellular Potential (ϕ_i) with D1

(b) Intracellular Potential (ϕ_i) with P1



(a) Extracellular Potential (ϕ_e) with D1

(b) Extracellular Potential (ϕ_e) with P1



(a) Gating variable (w) with D1

(b) Gating variable (w) with P1

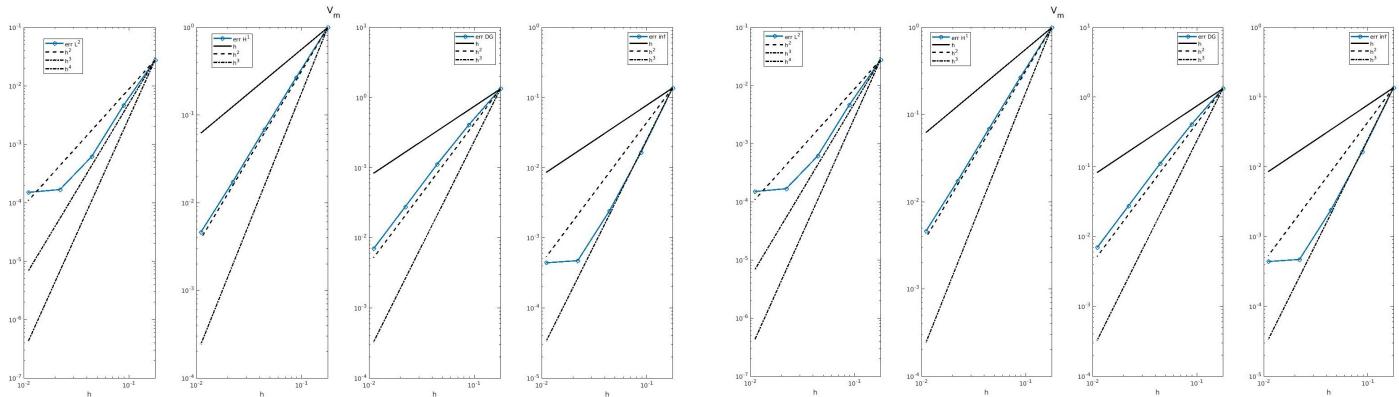
6.2.2 P2-D2

For what concerns the second order polynomials, we achieve slightly different results. Indeed, we notice two unexpected facts: first of all ϕ_i and ϕ_e errors trends are not identical when we adopt the two basis. However, this is not a huge inconsistency as it is simply due to the uniqueness imposition that have different effects for Dubiner and FEM. Indeed, the differences are visible only in the L^2 and L^∞ errors and, moreover, these differences are eliminated when potentials subtract to get V_m .

The second unexpected result is the flatter segment that some errors trends have for small mesh sizes. This is probably due to the influence of other cause of errors, especially the time-discretization errors, when the space discretization errors become very small. This is the reason why this effect was not present for $D1 - P1$: in the previous case, space errors were still too big and other causes of error negligible.

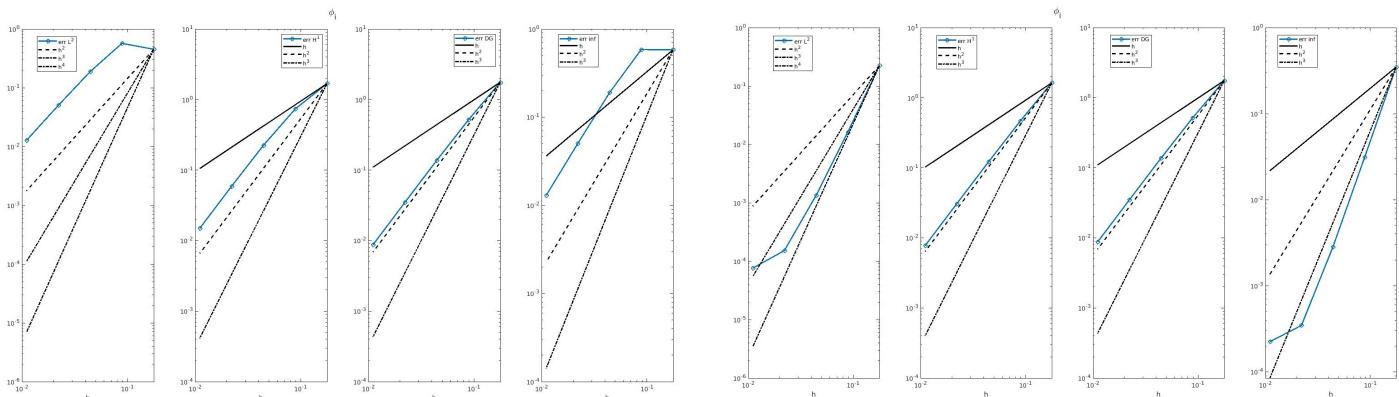
We then underline that these two unexpected effects are not due to the Dubiner discretization implementation itself. The first difference is due to the coefficient imposition that can be improved with a mean value imposition as seen in Section (5.2). The second fact is due to time discretization errors that can be reduced if we simply reduce the time-step. Moreover, if this happens, it means that space discretization errors are very small (in general, a positive fact). This is why we neglect these effects for the error order estimations and we can state that Dubiner method goodness keeps intact.

In conclusion, if we do not care so much about these two phenomena, plots show a second order for H^1 and L^2 errors and third order for L^2 and L^∞ errors.



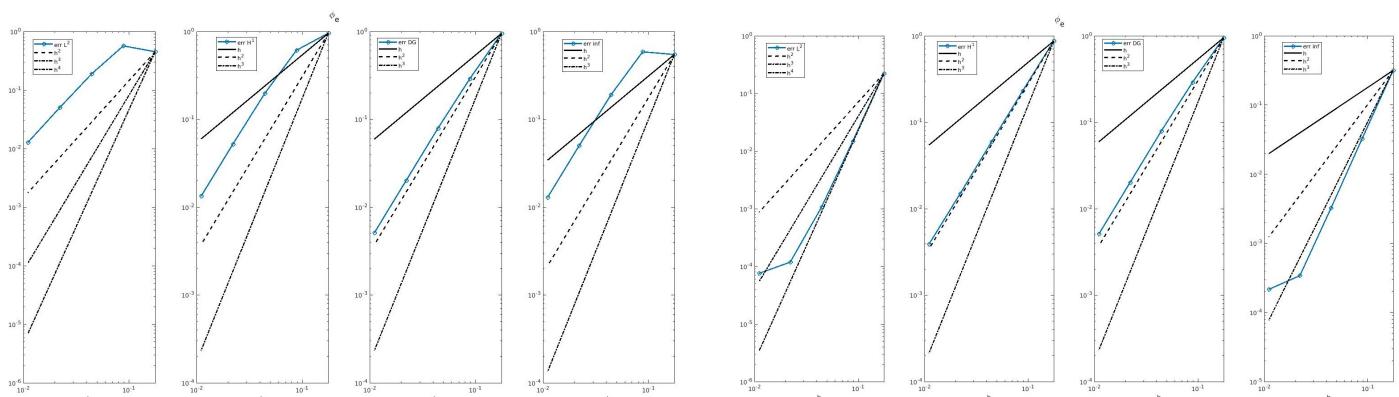
(a) Difference of potential (V_m) with D2

Difference of potential (V_m) with P2



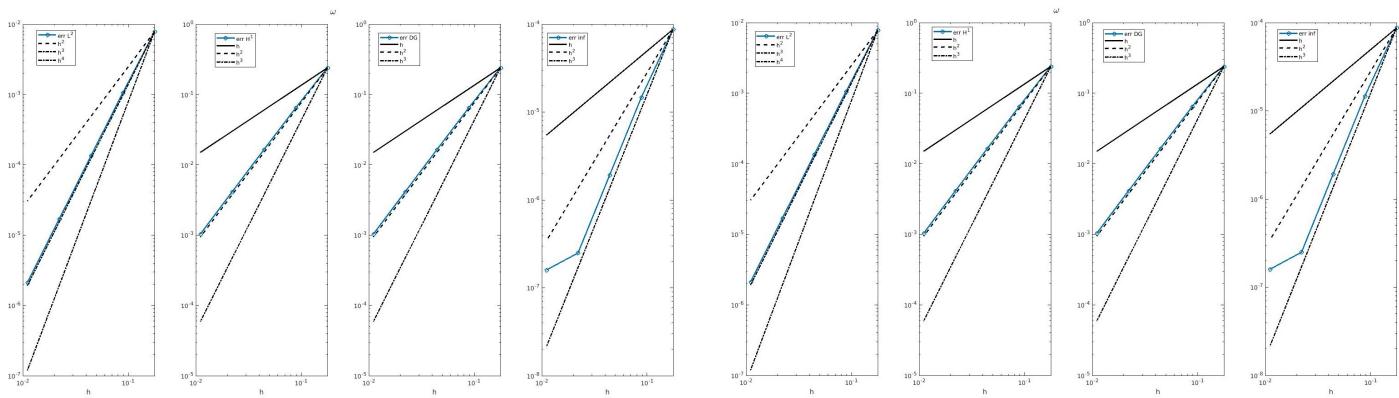
(a) Intracellular Potential (ϕ_i) with D2

(b) Intracellular Potential (ϕ_i) with P2



(a) Extracellular Potential (ϕ_e) with D2

(b) Extracellular Potential (ϕ_e) with P2

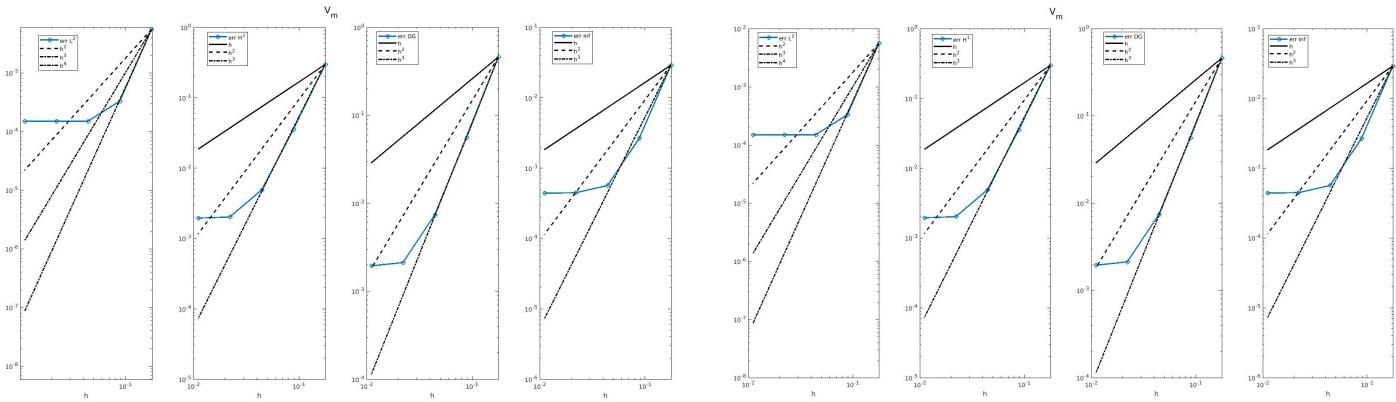
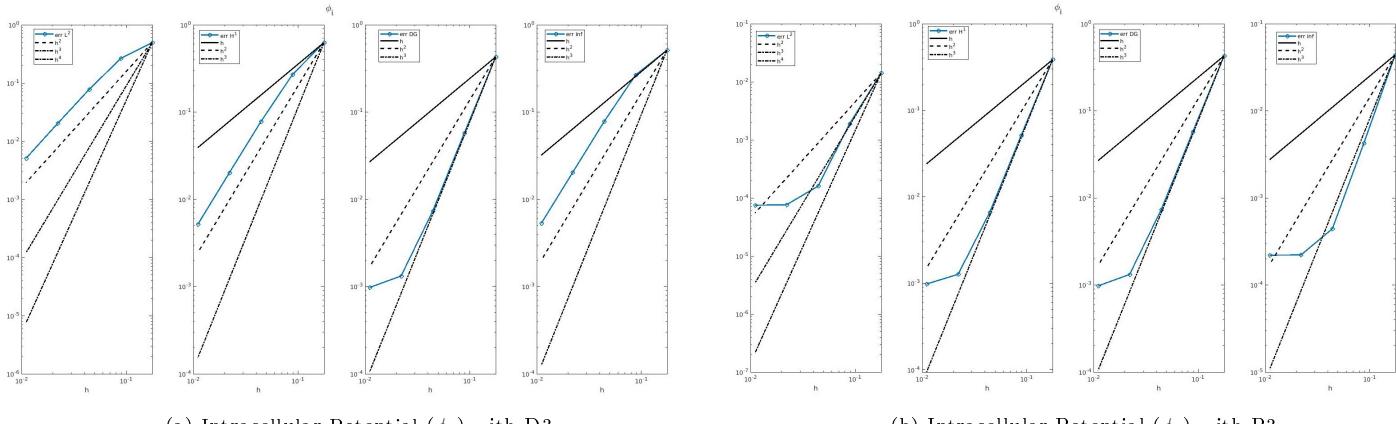
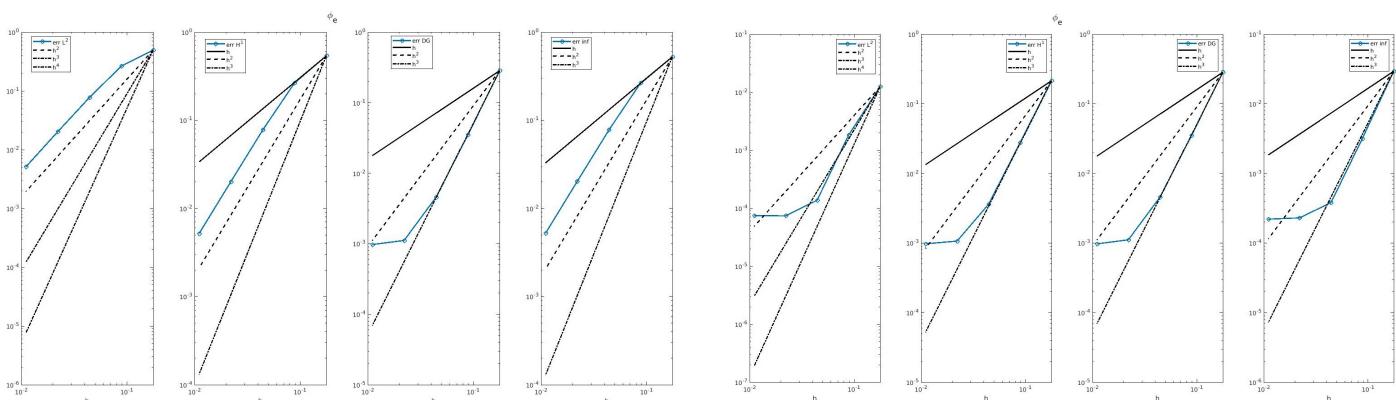
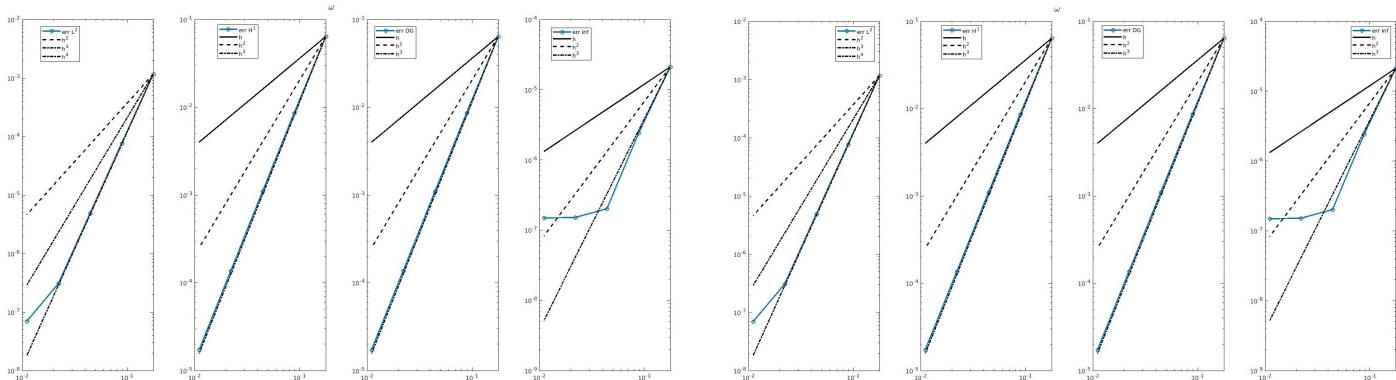


(a) Gating variable (w) with D2

(b) Gating variable (w) with P2

6.2.3 P3-D3

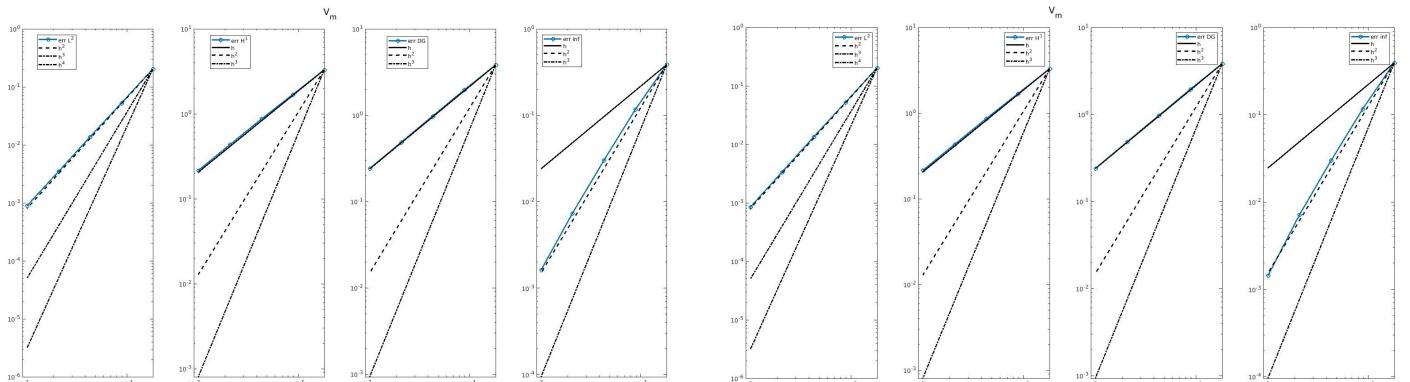
Finally, for polynomials of degree 3, we still find expected results except for the two phenomena already discussed in Section (6.2.2). However, because of the third order precision, space errors are smaller and then these effects amplified. If we do not consider them, we still get expected errors orders that are third for H^1 and DG errors and fourth order for L^2 and L^∞ .

(a) Difference of potential (V_m) with D3(b) Difference of potential (V_m) with P3(a) Intracellular Potential (ϕ_i) with D3(b) Intracellular Potential (ϕ_i) with P3(a) Extracellular Potential (ϕ_e) with D3(b) Extracellular Potential (ϕ_e) with P3(a) Gating variable (w) with D3(b) Gating variable (w) with P3

6.3 Comparison between different time discretization methods

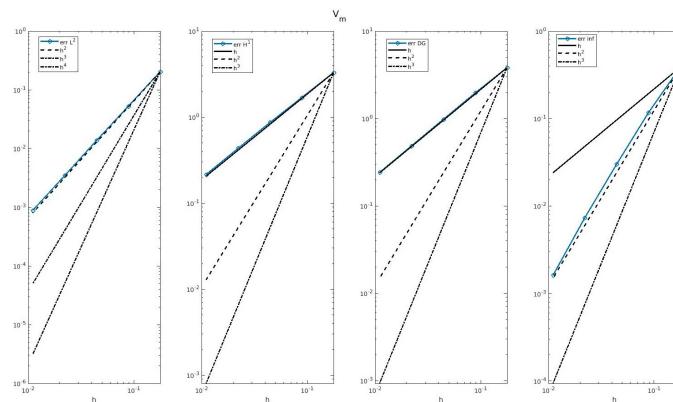
In Section (4) three methods for the time discretization have been proposed and discussed. An error analysis comparison has been executed and gives the following graphical results.

It is clear that the three methods share an identical error trend. This is expected since they were initially defined with the same order. The main difference between the three is the explicit/implicit choice for the different terms that, in general, gives different results only in terms of stability and not of order. Moreover, it is noteworthy to remember that non-linear term is considered as explicit for all the methods and then no temporal strategy is completely implicit. For this reason, it may be that the three methods have very similar behavior under the stability aspect too.

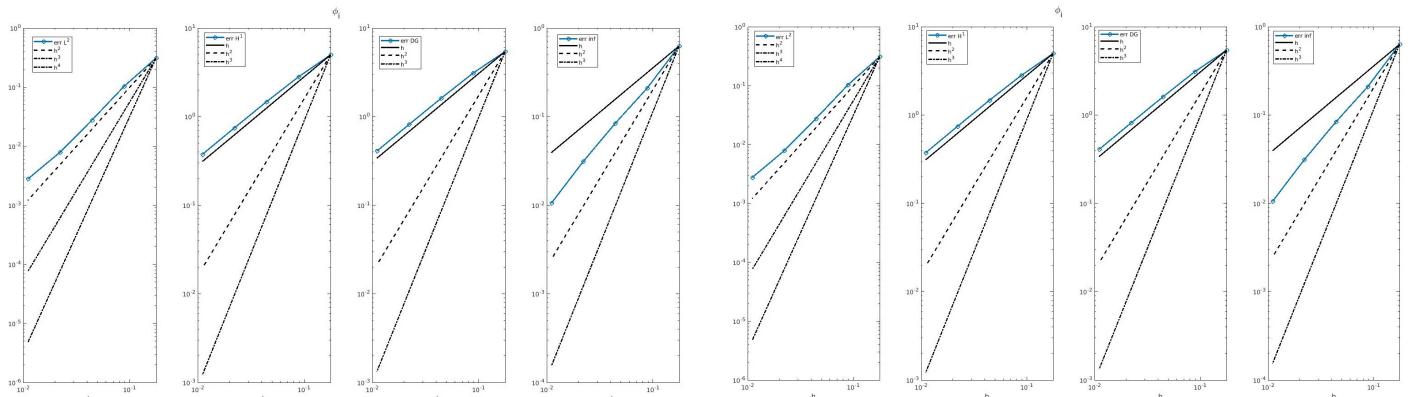


V_m with D1 - Semi-implicit method

V_m with D1 - Godunov operator-splitting method

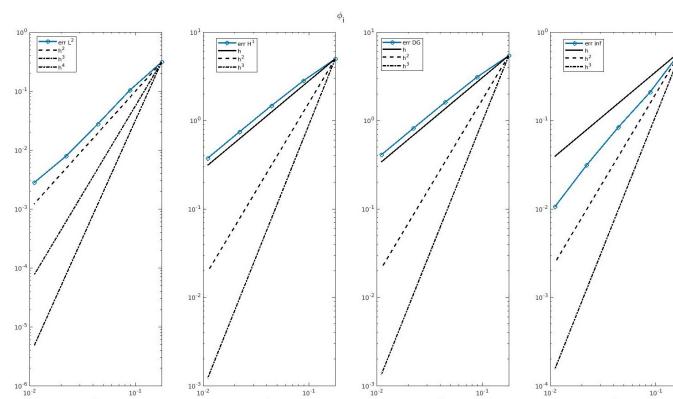


V_m with D1 - Quasi-implicit operator-splitting method

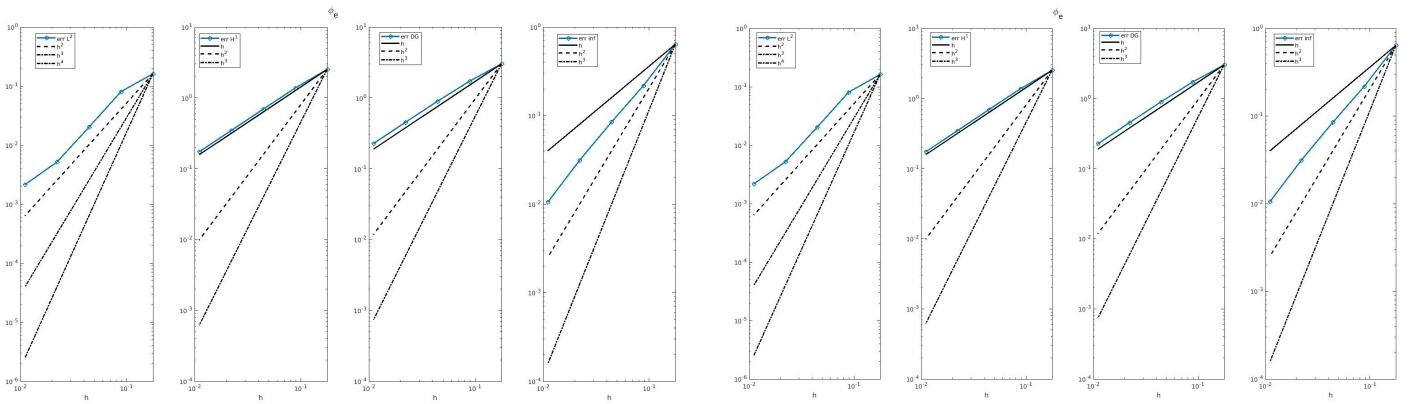


ϕ_i with D1 - Semi-implicit

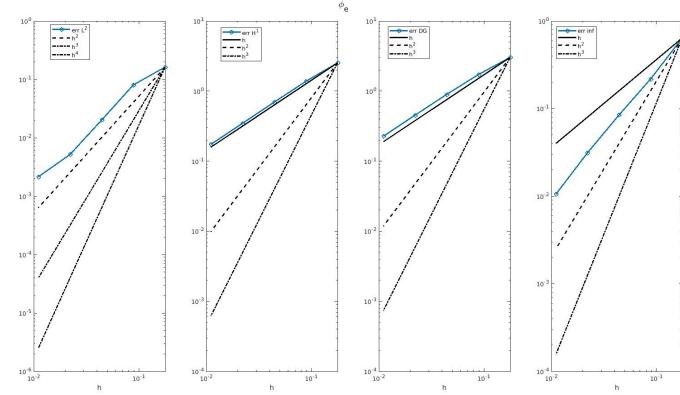
ϕ_i with D1 - Godunov operator-splitting method



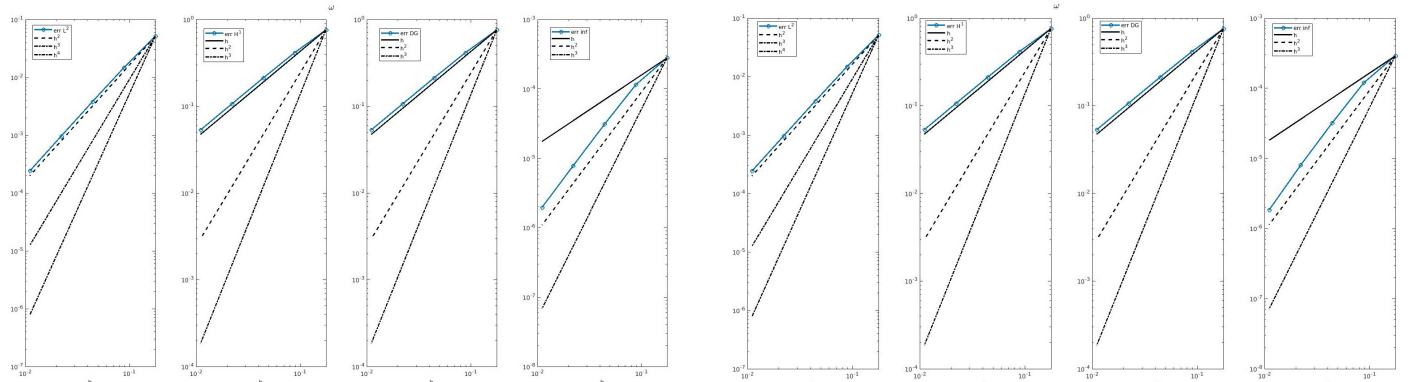
ϕ_i with D1 - Quasi-implicit operator-splitting method



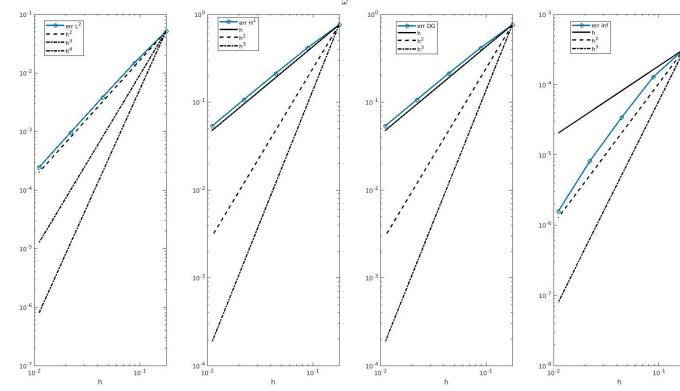
ϕ_e with D1 - Godunov operator-splitting method



ϕ_e with D1 - Quasi-implicit operator-splitting method



w with D1 - Godunov operator-splitting method



w with D1 - Quasi-implicit operator-splitting method

6.4 Comparison between methods for uniqueness of ϕ_i and ϕ_e

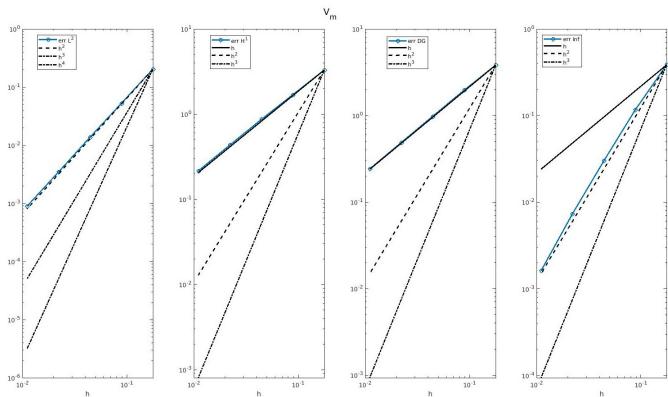
Referring to the Section (5), we have presented and explained two different methods to impose uniqueness of the cellular potentials. The simplest one, adopted for the previous error analysis, imposes the value of the function in a specific point. On the other hand, the second one imposes the mean value at zero.

First of all, we notice that the behavior for V_m and w are the same. This fact follows the theory since the imposition method affects only the values of the two potentials. On the other hand, for what concerns ϕ_i and ϕ_e , we see very similar results (even if not identical, the second method seems slightly more regular in the L^2 norm). Indeed, as already anticipated in Section (5.2.4), problems that does not imply very ill-conditioned systems can be solved in both ways with very small differences. Our proposed test case belongs to this family and for this reason we adopted the first method for all the previous error analysis studies as it is almost equivalent to the second one. These plots, then, confirm that the two methods are consistent and, most of the times, equivalent.

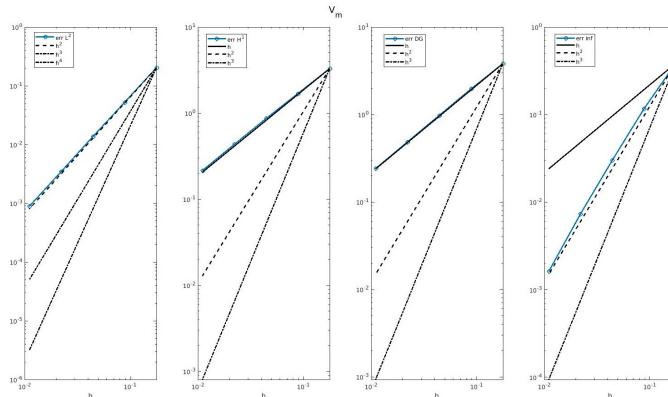
On the other hand, it has already been stated that some peculiar problems can be solved only using the second method. Indeed, there could be huge differences for the solution or the system may even be not solvable because of a null, or almost null, determinant. These problems will be considered in Section (7) about physical simulations. However, since there is no known analytical solution for these problems, we cannot make now an error analysis like the previous one.

In conclusion, we successfully imposed the potentials uniqueness directly into the system thanks to these two strategies. For what concerns our two initial objectives (Section (5.2)), we achieved both them since:

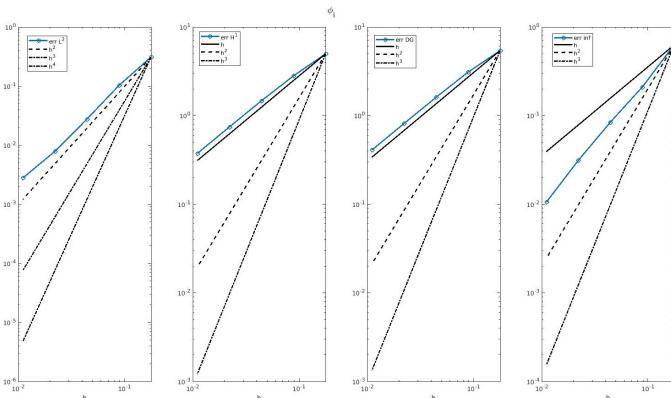
- Error analysis studies demonstrate that ϕ_i and ϕ_e potentials converge to the exact potentials as V_m and w do. Moreover, plots show that all the error orders are clearly defined.
- Conditional numbers turn to be considerably decreased. Our measures proved that they passed from $\approx 10^{17}$ to $\approx 10^7$ for both the strategies.



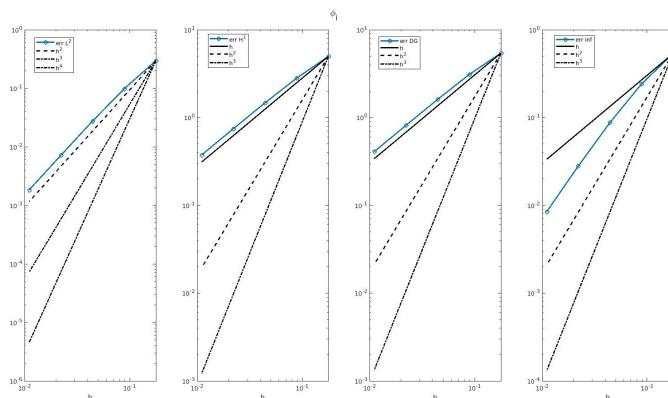
V_m with D1 imposing the function in a specific point



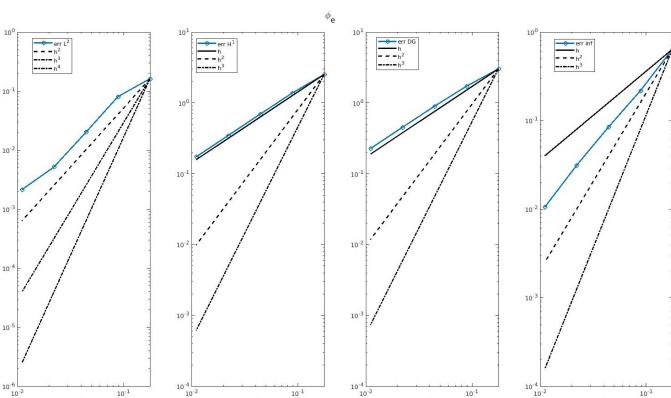
V_m with D1 imposing the zero mean



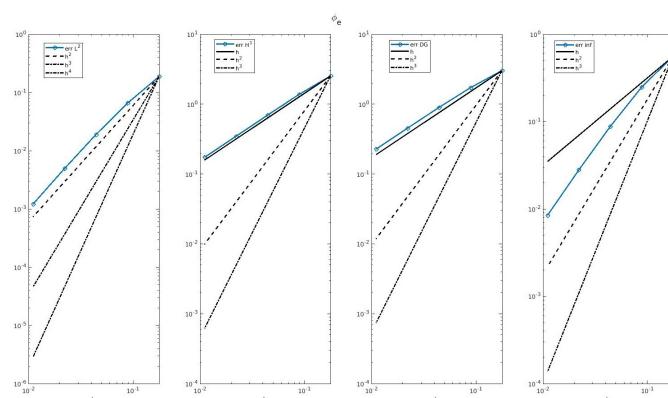
ϕ_i with D1 imposing the function in a specific point



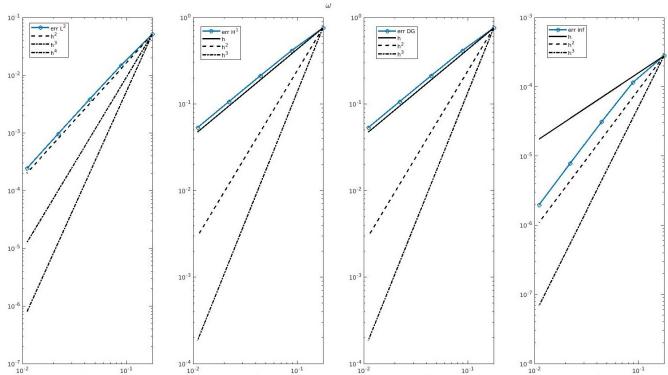
ϕ_i with D1 imposing the zero mean



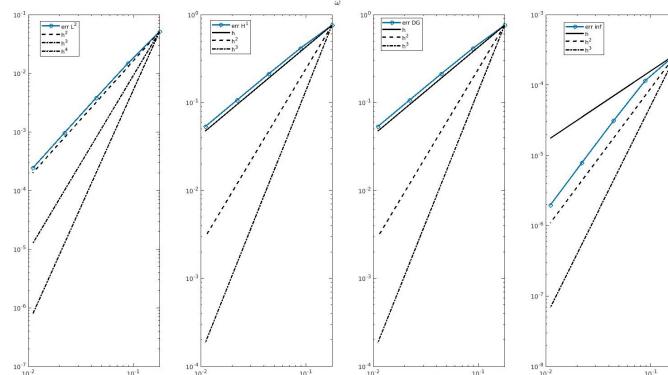
ϕ_e with D1 imposing the function in a specific point



ϕ_e with D1 imposing the zero mean



w with D1 imposing the function in a specific point



w with D1 imposing the zero mean

7 Realistic simulations

Our final goal is to exploit strategies and tools presented in the previous sections to run simulations that can get closer to the cardiac electrophysiology phenomena. It is noteworthy to warn that no exact solutions are known for these kinds of problems and what we care about is only visual. Moreover, we warn that there are some limitations to the complete resemblance to the realistic phenomena, for instance the domain shape. Our general choices for all the test-cases are the following:

Table 2: Parameters for pseudo-realistic simulations

Domain	$\begin{bmatrix} -0.025 & 0.035 \\ -0.025 & 0.035 \end{bmatrix}$
Temporal scheme	Semi-implicit
Polynomials space	D1
dt	0.0001
$nREF$	5
Initial condition for V_m	0
Initial condition for w	0
I_i^{ext}	$I \cdot 10^3 \chi_{[0.001,0.002]}(t) \chi_{[0.0045,0.0055]}(x) \chi_{[0.0045,0.0055]}(y)$
I_e^{ext}	$I \cdot 10^3 \chi_{[0.001,0.002]}(t) \chi_{[0.0045,0.0055]}(x) \chi_{[0.0045,0.0055]}(y)$
b_i	0
b_e	0
χ_m	10^5
C_m	10^{-2}
Σ_i	$\begin{bmatrix} 0.34 & 0 \\ 0 & 0.06 \end{bmatrix}$
Σ_e	$\begin{bmatrix} 0.62 & 0 \\ 0 & 0.24 \end{bmatrix}$

Where I is a positive value to be chosen depending on the context. We remind that the mean value imposition is always chosen for realistic simulations. Moreover, we observe

that the main differences with the previous parameters are the square domain that is no more unitary and the anisotropy of the diffusion tensors. The latter choice is motivated from the real utility of the *Bidomain Model* if compared to the *Monodomain Model*, where the two tensors are assumed to be equal or proportional. Parameters values are taken as usual from literature, especially from [11].

Physically, this setting represents a square section of the surface of the heart that is electrically isolated (homogeneous boundary conditions) and an external current that is applied in a central region for a limited interval of time.

To conclude, observe that the compatibility condition for the existence of the solution is satisfied since both boundary conditions and external currents are the same in the intracellular and extracellular regions.

For what concerns *FitzHugh-Nagumo* parameters, there are no general values as for the previous ones. To generalize this aspect, we defined two different test-cases with two different sets of parameters, namely:

Table 3: FitzHugh-Nagumo Parameters for pseudo-realistic simulations

	Test-case 1	Test-case 2
k	19.5	1
ϵ	1.2	0.2232
γ	0.1	4.0322
a	$13 \cdot 10^{-3}$	0.004

Basically, the first test-case comes from the values already adopted for Section (6) and taken from past projects [4],[2], [9]. On the other hand, second test-case comes from [1].

7.1 First test-case

For both the test-cases, we aim to show two different situations depending on the external current intensity: firstly, for too weak currents, the electrical activation should miss and this implies that the potentials are not capable to hold up. Secondly, if the intensity is over a certain threshold, we should see the electrical activation and the resulting diffusion. Several simulations have proved that:

- $I = 500 \cdot 10^3$ is a suitable value for the *underflow* case.
- $I = 600 \cdot 10^3$ is a suitable value for the *overflow* case.

Visual results are the following:

Figure 25: Test-case 1, missed activation ($I = 500 \cdot 10^3$)

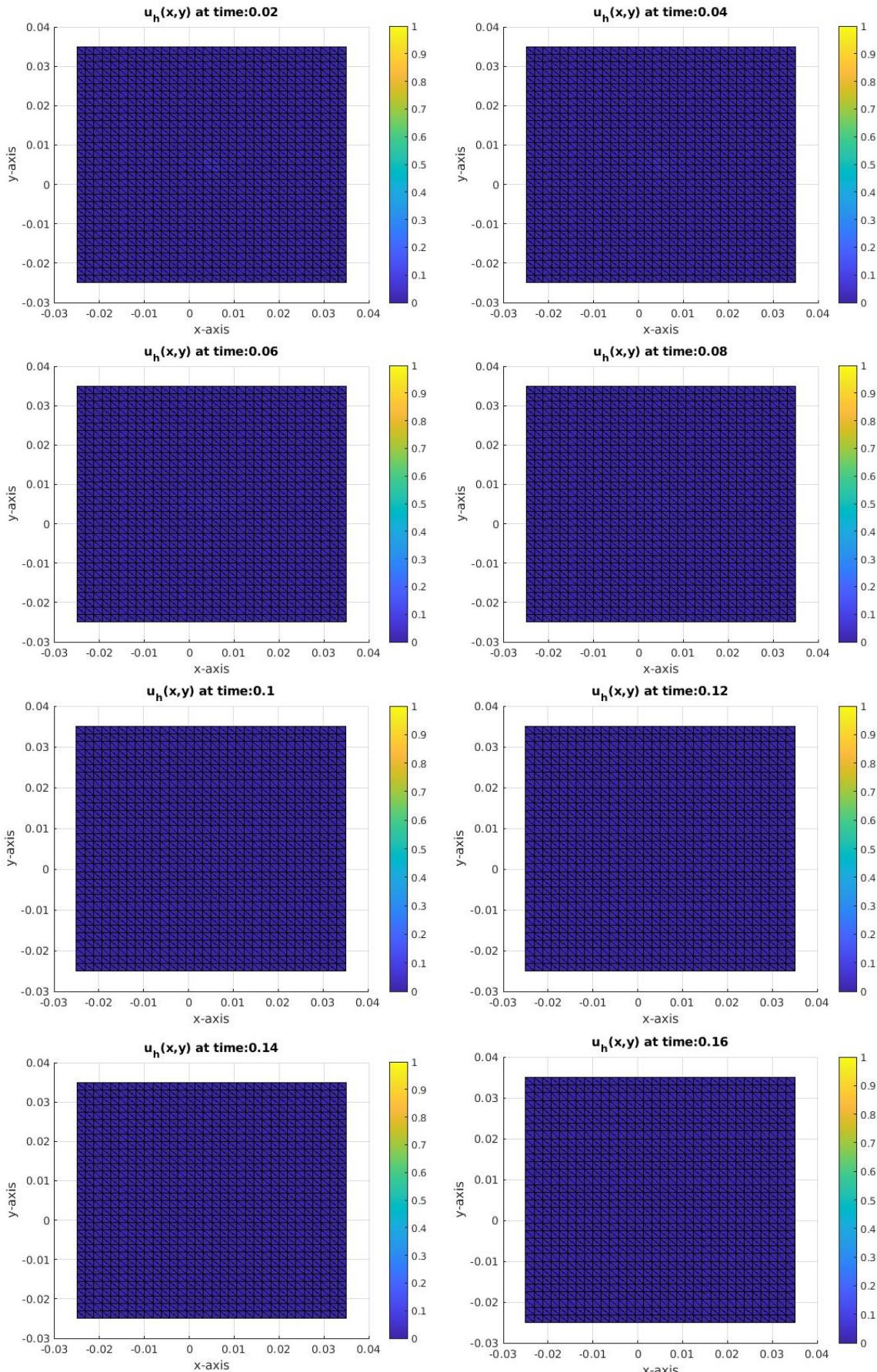
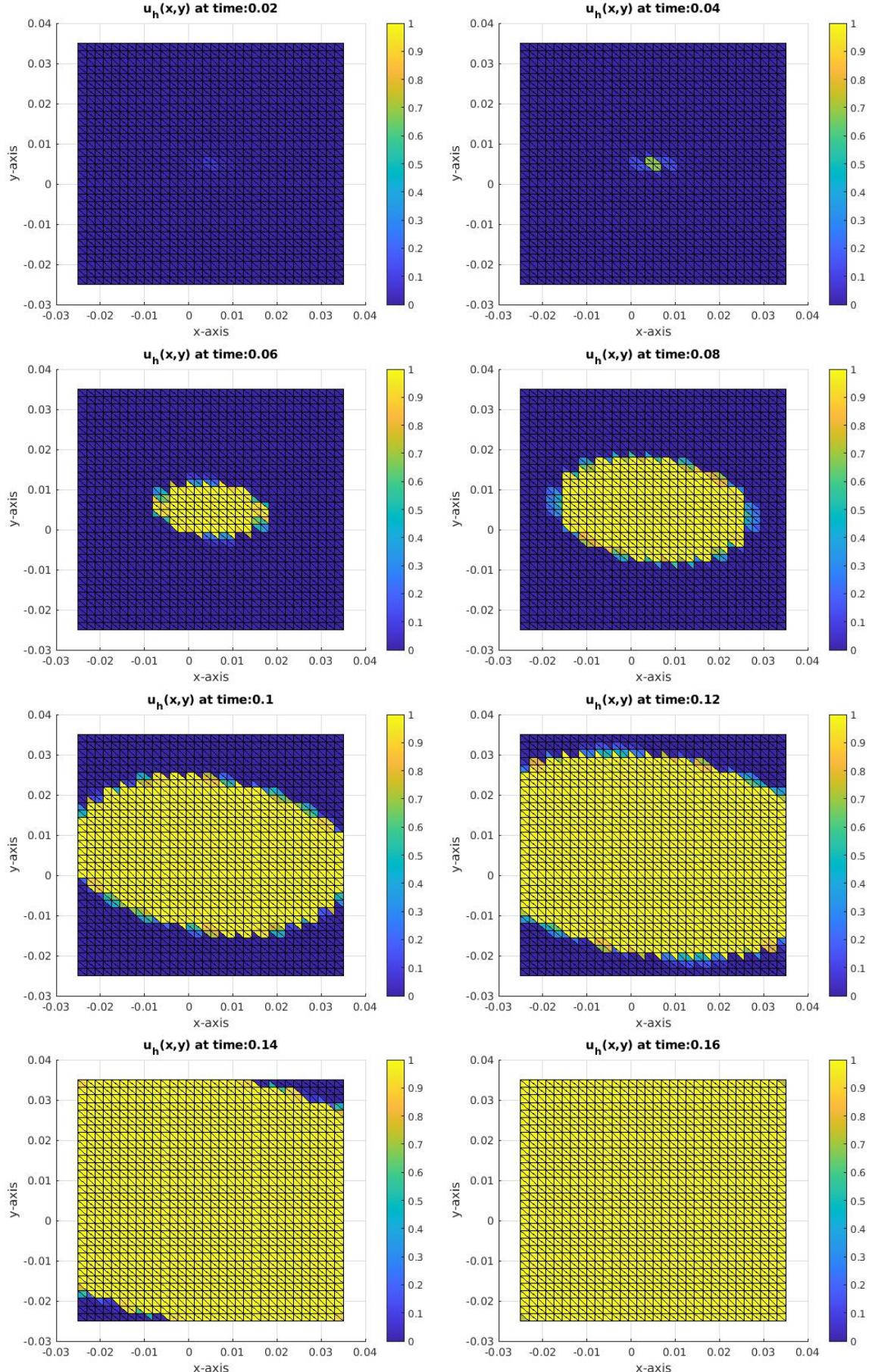


Figure 26: Test-case 1, activation ($I = 600 \cdot 10^3$)



7.2 Second test-case

For these modified parameters, the threshold for the activation is different as we can state that:

- $I = 15000 \cdot 10^3$ is a suitable value for the *underflow* case.
- $I = 20000 \cdot 10^3$ is a suitable value for the *overflow* case.

Figure 27: Test-case 2, missed activation ($I = 15000 \cdot 10^3$)

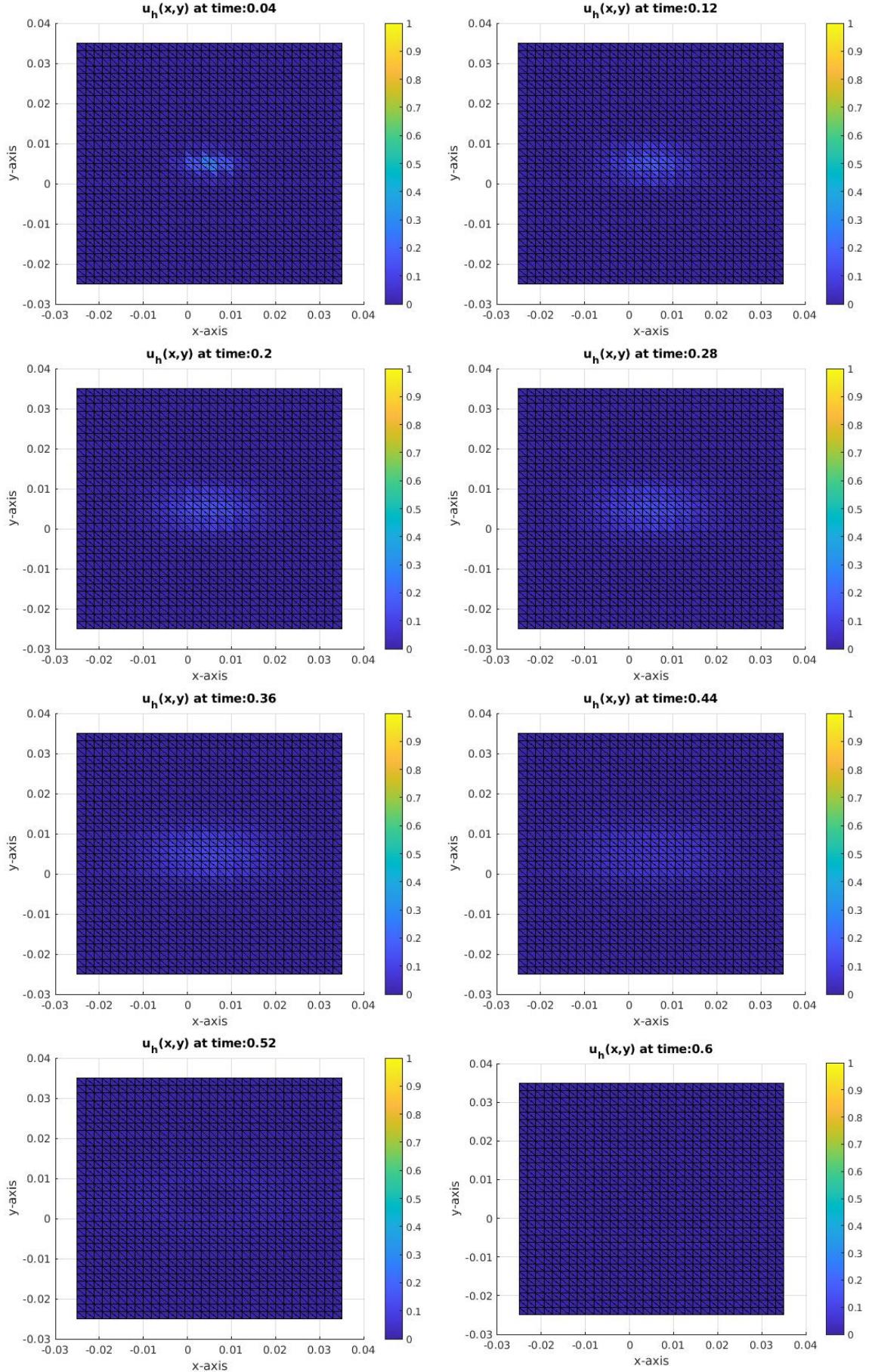
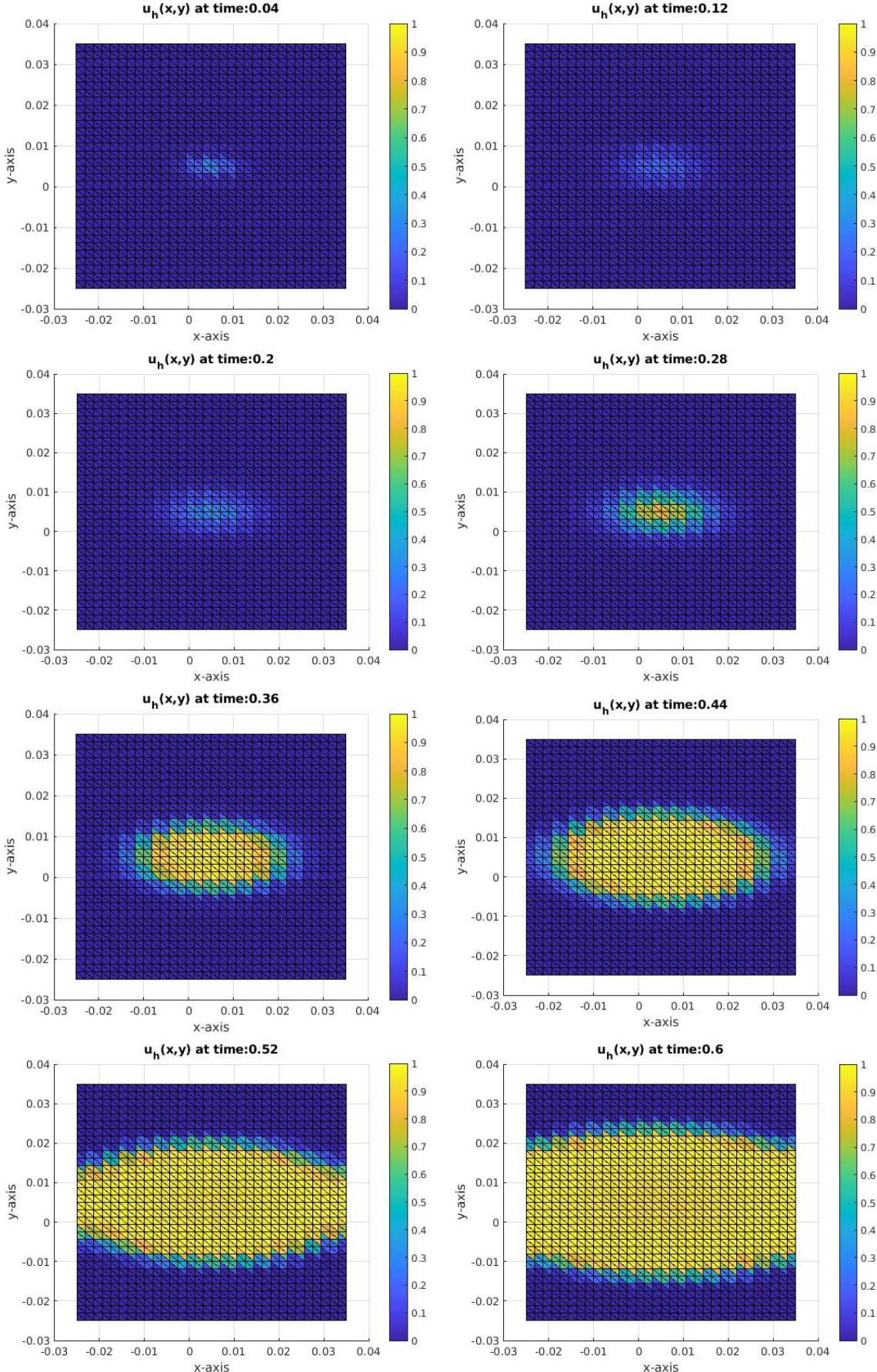


Figure 28: Test-case 2, activation ($I = 20000 \cdot 10^3$)



8 Numerical Codes

8.1 Implementation: Dubiner to FEM

```

1 function [ u0 ] = dubiner_to_fem ( uh , femregion , Data )
2
3 ...
4 ...
5 ...
6
7 u0 = zeros(femregion.ndof ,1);
8
9 % loop over all the elements
10 for ie = 1:femregion.ne
11
12 % to get the global indexes for the nodes of ie
13 nln = femregion.nln;
14 index = (ie-1)*nln*ones(nln,1) + [1:nln]';
15
16 for i = 1 : nln
17     for j = 1 : nln
18         u0(index(i)) = u0(index(i)) + uh(index(j))*phi(1,i,j);
19     end
20 end
21 end
```

8.2 Implementation: FEM to Dubiner

```

1 function [ u0 ] = fem_to_dubiner ( uh , femregion , Data )
2
3 ...
4 ...
5 ...
6
7 u0 = zeros(femregion.ndof ,1);
8
9 % loop over all the elements
10 for ie = 1:femregion.ne
11
12 % to get the global indexes for the nodes of ie
13 nln = femregion.nln;
14 index = (ie-1)*nln*ones(nln,1) + [1:nln]';
15 % loop over local degrees of freedom
16 for i = 1 : nln
17     % loop over 2D quadrature points
18     for k = 1:length(w_2D)
19         uh_eval_k = 0;
20         % loop to evaluate uh in a quadrature point
21         for j = 1 : nln
22             uh_eval_k = uh_eval_k + uh(index(j))*phi_fem(1,k,j);
```

```

23     end
24     u0(index(i))=u0(index(i))+uh_eval_k*phi_dub(1,k,i).*w_2D(k);
25   end
26 end

```

8.3 Implementation: Semi-implicit method

```

1 MASS = [M -M; -M M];
2 ZERO = sparse(length(M), length(M));
3 MASS_W = [M ZERO; ZERO -M];
4 STIFFNESS = [Ai ZERO; ZERO Ae];
5
6 for t=dt:dt:T
7
8   w1 = 1/(1+epsilon*gamma*dt)*(w0+epsilon*dt*Vm0);
9   w1=cat(1,w1, w1);
10  Vm0 = cat(1,Vm0, Vm0);
11
12  fi = assemble_rhs_i(femregion, neighbour, Data, t);
13  fe = assemble_rhs_e(femregion, neighbour, Data, t);
14  f1 = cat(1, fi, fe);
15
16  [C] = assemble_nonlinear(femregion, Data, Vm0);
17  NONLIN = [C -C; -C C];
18
19  r = f1 + ChiM*Cm/dt * MASS_W * Vm0 - ChiM * MASS_W * w1;
20
21  B=ChiM*Cm/dt * MASS + (STIFFNESS + NONLIN);
22
23  u1 = B \ r;
24
25  f0 = f1;
26  Vm0 = u1(1:11)-u1(11+1:end);
27  u0 = u1;
28  w0 = w1(1:11);
29 end

```

8.4 Implementation: Godunov operator-splitting method

```

1 ZERO = sparse(11, 11);
2 MASS = (ChiM*Cm/dt)*[M, -M; M -M];
3 MASSW = ChiM*[M, ZERO; ZERO, M];
4
5 for t=dt:dt:T
6
7   fi = assemble_rhs_i(femregion, neighbour, Data, t);
8   fe = assemble_rhs_e(femregion, neighbour, Data, t);
9   f1 = cat(1, fi, -fe);

```

```

10 [C] = assemble_nonlinear(femregion, Data, Vm0);
11
12 w1 = (1 - epsilon*gamma*dt)*w0 + epsilon*dt*Vm0;
13 B = MASS + [Ai, ZERO; ZERO, -Ae];
14 r = -MASSW*[w0; w0] + ((Cm/dt)*MASSW - [C, ZERO; ZERO, C])
15 * [Vm0; Vm0] + f1 ;
16
17 Vm0 = u1(1:11) - u1(11+1:end);
18 u0 = u1;
19 w0 = w1;
20
21 end

```

8.5 Implementation: Quasi-implicit operator-splitting method

```

1 ZERO = sparse(11, 11);
2
3 for t=dt : dt : T
4
5 [C] = assemble_nonlinear(femregion, Data, Vm0);
6 Q = (ChiM*Cm/dt)*M + C - (epsilon*ChiM*dt)/(1+epsilon*gamma*dt)*M;
7 R = (ChiM*Cm/dt)*M*Vm0 - (ChiM)/(1+epsilon*gamma*dt)*M*w0;
8
9 fi = assemble_rhs_i(femregion, neighbour, Data, t);
10 fe = assemble_rhs_e(femregion, neighbour, Data, t);
11 f1 = cat(1, fi, -fe);
12
13 B = [Q, -Q; Q, -Q] + [Ai, ZERO; ZERO, -Ae];
14 r = [R; R] + f1;
15
16 u1 = B \ r;
17
18 Vm1 = u1(1:11) - u1(11+1:end);
19
20 w1 = (w0 + epsilon*dt*Vm1)/(1+epsilon*gamma*dt);
21
22 f0 = f1;
23 Vm0 = u1(1:11) - u1(11+1:end);
24 u0 = u1;
25 w0 = w1;
26
end

```

8.6 Implementation: imposing a value in a specific point

```

1 function [A, b] = assign_phi_i(A, b, t, Data, femregion)
2
3 ...
4 ...
5 ...

```

```

6
7 if (Data.fem(1) == 'P')
8
9 x = femregion.dof(1,1); % x-coordinate of the first dof point
10 y = femregion.dof(1,2); % y-coordinate of the first dof point
11 exact_coeff = eval(Data.exact_sol_i); % evaluation of exact sol
12
13
14 elseif (Data.fem(1) == 'D')
15 x0 = femregion.dof(3,1); % bottom-left corner of the first element
16 y0 = femregion.dof(3,2);
17 h = femregion.dof(1,1) - femregion.dof(3,1); % length of the element
18
19 exact_coeff = 0;
20 index = 1;
21
22 ...
23 ...
24 ...
25
26 % the first coeff is the L2 scalar product of uh with the first
27 % basis function. To get the right coeff, we compute the scalar
28 % product between the exact solution and the first basis function
29
30 for k = 1:length(w_2D) % loop over 2D quadrature points
31     % physical coords of the integration point
32     x = x0 + h*node_2D(k,1);
33     y = y0 + h*node_2D(k,2);
34     evalsol = eval(Data.exact_sol_i);
35     exact_coeff = exact_coeff + evalsol*phi_dub(1,k,index).*w_2D(k);
36 end
37
38 end
39
40
41 % we change the system coefficients to impose u(1) = exact_coeff
42 Nh = length(b);
43 b = b - A(:,1)*exact_coeff;
44 b(1) = exact_coeff;
45 A(:,1) = zeros(Nh,1);
46 A(1,:) = zeros(1,Nh);
47 A(1,1) = 1;

```

8.7 Implementation: imposing zero mean

```

1 function [A, b] = assign_null_average (A, b, Data, femregion)
2
3 Nh = length(b)/2;
4
5 if (Data.fem(1) == 'P')

```

```

6
7     ...
8     ...
9     ...
10
11    for k = 1:length(w_2D)
12        coeff = coeff + phi_fem(1,k,1).*w_2D(k);
13    end
14
15    for i = 1:Nh
16        A(i,2*Nh+1)=coeff ;
17        A(2*Nh+1,i)=coeff ;
18    end
19
20
21 elseif (Data.fem(1)=='D')
22
23     ...
24     ...
25     ...
26
27    for p = 1:femregion.nln
28        p_int = 0;
29        for k = 1:length(w_2D)
30            p_int = p_int + phi_dub(1,k,p).*w_2D(k);
31        end
32        coeff(p)=p_int ;
33    end
34
35    for i = 1:femregion.nln:Nh
36        A(2*Nh+1,i:i+femregion.nln-1)=coeff';
37        A(i:i+femregion.nln-1,2*Nh+1)=coeff;
38    end
39
40 end
41
42 b = [b;0];

```


References

- [1] V. Anaya et al. “A Virtual Element Method for a Nonlocal FitzHugh-Nagumo Model of Cardiac Electrophysiology”. In: *IMA Journal of Numerical Analysis* 40 (2020), pp. 1544–1576.
- [2] F. Andreotti and D. Uboldi. *Discontinuous Galerkin approximation of the monodomain problem*. Politecnico di Milano, 2021.
- [3] P. F. Antonietti and P. Houston. “A Class of Domain decomposition Preconditioners for hp-Discontinuous Galerkin Finite Element Methods”. In: *Journal of Scientific Computing* 46 (2011), pp. 124–149.
- [4] M. Bagnara. *The Inverse Potential Problem of Electrocardiography Regularized with Optimal Control*. Politecnico di Milano, 2020.
- [5] Y. Bourgault, Y. Coudière, and C. Pierre. “Existence and uniqueness of the solution for the bidomain model used in cardiac electrophysiology”. In: *Nonlinear Analysis: Real World Applications* 10 (2009), pp. 458–482.
- [6] P. Colli Franzone, L. F. Pavarino, and S. Scacchi. *Mathematical Cardiac Electrophysiology*. Cham: Springer-Verlag, 2014.
- [7] M. Dubiner. “Spectral Methods on Triangles and Other Domains”. In: *Journal of Scientific Computing* 6 (1991), pp. 345–390.
- [8] A. Ferrero, F. Gazzola, and M. Zanotti. *Elementi di analisi superiore per la fisica e l'ingegneria*. Bologna: Società editrice Esculapio, 2015.
- [9] L. Marta and M. Perego. *Discontinuous Galerkin approximation of the bidomain system for cardiac electrophysiology*. Politecnico di Milano, 2021.
- [10] A. Quarteroni. *Modellistica Numerica per Problemi Differenziali*. Milan: Springer-Verlag, 2016.
- [11] A. Quarteroni, A. Manzoni, and C. Vergara. “The cardiovascular system: Mathematical modelling, numerical algorithms and clinical applications”. In: *Acta Numerica* (2017), pp. 365–590.
- [12] S. Salsa. *Equazioni a derivate parziali*. Milan: Springer-Verlag, 2016.
- [13] S. J. Sherwin and G. E. Karniadakis. “A new triangular and tetrahedral basis for high-order finite element methods”. In: *International journal for numerical methods in engineering* 38 (1995), pp. 3775–3802.
- [14] R. Spiteri and S. Torabi Ziaratgahi. “Operator splitting for the bidomain model revisited”. In: *Journal of Computational and applied Mathematics* 296 (2016), pp. 550–563.