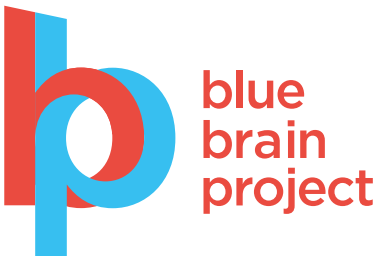

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

EPFL Semester Project – Spring 2016

Explicit stabilized methods for numerical integration of NEURON's equations

Author:
Nicolò Ripamonti
Sciper n. 260853

Supervisors:
Casalegno Francesco
Rosilho de Souza Giacomo

Responsible:
Prof. Abdulle Assyr

June 2, 2016

Contents

Chapter 1

Introduction

The neuron is the basic working unit of the nervous system and group of neurons are connected into neural networks. They control, through electrical and chemical signals, different processes: muscular contractions, sensory responses and glandular secretions are among the most important and are described in details in [13]. Moreover, neural networks structures evolve, strengthening or weakening their connections over time throug synapses, affecting the *synaptic plasticity*, responsible of learning and memory. Hence, understanding this communication mechanism is crucial.

As years passes, computer simulations are becoming a powerful tool to investigate the neuron behaviour, even by considering large and ramified networks. One of the most flexible and powerful simulators available is NEURON [4], developed by Michael Hines at Yale university. NEURON is actually employed by different researchers around the world to study complex neural networks and how they affect human life. More recently NEURON has proven to be valid to understand the mechanisms behind neurological disorders [2] [3] [8], rising concerns for a world ageing population. A more in-depth knowledge of neuron structure will also lead to interesting results outside the biological scope. Since Moore's Law for traditional computer seems to be outdated in last years, new computing architectures are needed: neuromorphic engineering, with which a brain-like structure is used, could benefit from a progress in this field [18].

One of the main advantages of NEURON is the separation between biological modeling and mathematical/computational knowledge behind its implementation. It offers different uniform/nonuniform spatial discretizations, adaptive/static time discretizations and various time integrators and the user can swith between them without recoding everything from the scratch.

In this work we will analyze the performance of possible alternatives, introduced in [1] and [22], to the standard integrators used by NEURON. The main advantage of this new method is that it doesn't require to solve a linear system at each iteration, guaranteeing at the same time strong stability prop-

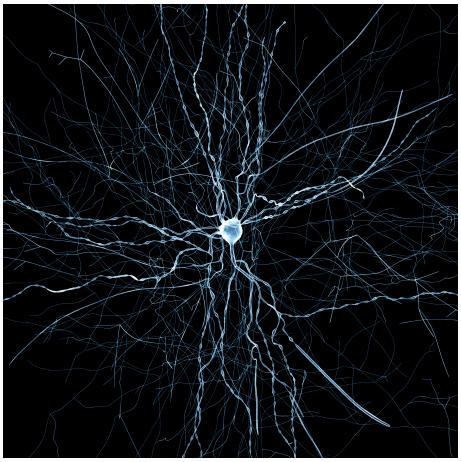

Figure 1.1: An image of a layer 2/3 large basket cell in a digital reconstruction of neocortical microcircuitry, taken from BBP website.

erty. In this work the computational domain will be a simplified branched dendrite. Even if it is not complex, it offers the possibility to study some interesting property, like potential propagation through a branching node: the diffusion, across the whole structure, of a single spike in the potential will be simulated. The variable of interests for the simulation will be the membrane potential and the ions gate states, which play an important regulation role, especially during the action potential.

This work was done with the support of the High Performance Computing (HPC) team of the Blue Brain Project (BBP). The BBP is a vanguard project with the final aim of digitally reconstructing the human brain. Their approach is to study the multilevel structure and function of the brain thanks to a detailed and immensive database of experimental data combined with super-computer based simulations. The different levels of simulation (molecules across the neuron membrane, cells and whole brain) are carried out by using different softwares, Neuron included, for which intense numerical studies and tests have been undertaken by the members of the team.

In the following chapters we will present the underlying Neuron mathematical model, originally developed in [12], to have at the end a set of equations that describe the problem. This will be followed by a chapter devoted to the particular discretization of the domain adopted in Neuron and then by a view over possible instability issues of the discretized scheme. In particular we will focus on the approach offered by [1] to deal with instability. In the last chapters, the results of our numerical simulation will be showed with some conclusions regarding the result of our work.

Chapter 2

Biological model

2.1 Structure and Physiology of the Neuron

A single neuron consists of three main parts (see Figure 2.1), each with its own peculiar functions [9]:

- The *soma*, which contains the cell nucleus and is responsible for RNA production.
- The *axons*, nerve fibers responsible of conducting impulses away from neuron cell to other cell in the body (output system).
- The *dendrites*, a branching outgrowth from a neuron capable of carrying external impulses from synapses to the soma (input system).

2.2 Equivalent Electrical Circuit : The Hodgkin–Huxley model

In 1952 Hodgkin and Huxley conducted a series of experiments with the aim of describing mathematically the behaviour of ions in a neuron during an action potential. The designed target was a squid giant axon, because it was big enough to use their particular and problem specific glass electrodes, at a time in which electron microscopes were not available yet. The informations regarding their discoveries and the presentation of the model can be found in the famous article [12], for which they had been awarded with a Nobel Prize.

Experimental results show that the process of depolarization of the surface membrane of a nerve fibre could be modeled as an equivalent electrical circuit (see Figure 2.2). It is important to underline that Hodgkin and Huxley derived their model not by describing the molecular events related to changes in permeability but they were driven by some questions regarding

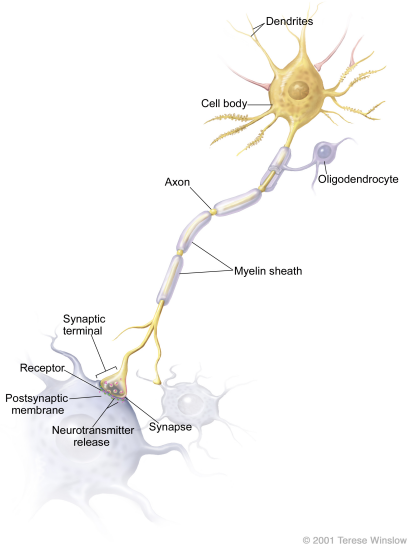

Figure 2.1: Neuron representation, taken from U.S. National Library of Medicine.

Figure 2.2: Equivalent electrical circuit

the general behaviour of cellular membrane[12]. Their starting assumption, verified later by data, was that the membrane permeability is influenced by the membrane potential and not by the membrane current: a direct consequence is that the main contribution to membrane permeability is the effect of electric fields on distribution of charged particles. This result shows the importance of including in the model the mechanism of the ions crossing the membrane (in particular sodium and potassium).

Even small changes in the potential are followed by a exponential increase in the ionic conductances of sodium and potassium: the effect of this phenomenon is of great importance since it affects the difference in ionic concentration between the inside of the cell and the extracellular liquid which in its turn generates a Nerst's potential that can be represented by a battery in the simplified circuit.

Another observed property is the behaviour of the membrane with respect to an imbalance between input and output current, i.e. more positive charges enter the cell than leave it or viceversa, which made them think that capacitive effects come into play to change membrane potential, which polarise or depolarise according to the change in the total charge. All these observations, combined with their previous hypothesis, were used to build a solid theoretical framework from which a set of equations could be developed.

2.3 Mathematical description: From current conservation to the cable equation

The total membrane current can be written as the sum of a capacity current (modeled as a perfect electrical capacitor without dielectric loss) and of an ionic current.

$$I = C_M \frac{\partial V}{\partial t} + I_i, \tag{2.1}$$

In (2.1) we used the following notation:

I is the total membrane current density [mA]

I_i is the ionic current density [mA]

V is the displacement of membrane potential from its resting value [mV]

C_M is the membrane capacity per unit area $[\frac{\text{mS}}{\text{cm}^2}]$

t is the time [ms]

The choice of model the currents in parallel is due to the similarity between ionic current when $\frac{dV}{dt} = 0$ and the capacity current when $I = 0$. This model is enriched by identifying the different contributions for the ionic current, as follows:

$$I_i = I_{\text{Na}} + I_{\text{K}} + I_{\text{L}},$$

where I_{Na} and I_{K} are the components related to different ions channels (Na=Sodium, K=Potassium) and I_{K} is the leak current. Using Ohm's law, expressed in term of conductance (i.e. $I = gV$), yields the equation,

$$I_i = g_{\text{Na}}(V - V_{\text{Na}}) + g_{\text{K}}(V - V_{\text{K}}) + g_{\text{L}}(V - V_{\text{L}}),$$

where V,V_{Na},V_{K} and V_{L} are the displacements from resting potentials. The sign of the individual ionic contribution can be positive or negative depending on the relation between the membrane voltage and the ionic potential: this can can lead to a problem of sign conventions, which is solved by setting positive a ionic current flowing out of the cell.

As said before, the conductance of sodium and potassium channel show a strong dependence on membrane potential, while g_{L} is usually taken as a constant: this is called leakage conductance and its value is smaller than the voltage-dependent conductances. Although Hodgkin and Huxley did not know the complex mechanism behind the rise and fall of ionic conductances, they were were able to reconstruct this relation by fitting their experimental results (see Figure (2.3)).

Without any first principles avalaible they observed that the potassium conductance is proportional to the fourth power of the solution of a first

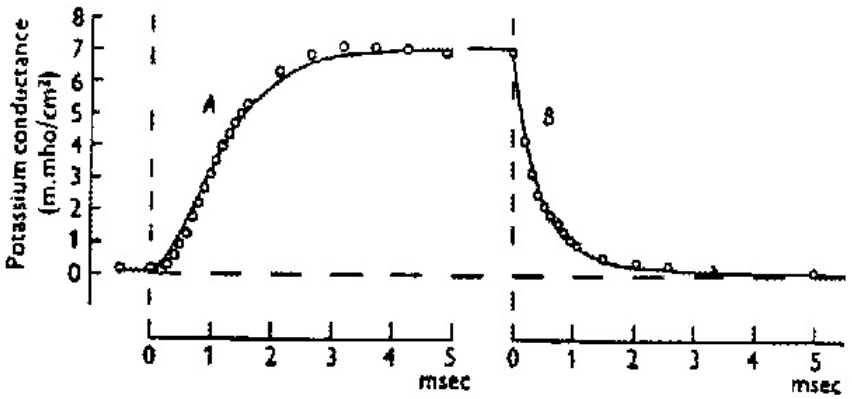

Figure 2.3: Rise and fall of potassium conductance associated with a depolarization and a polarization of 25 mV

order equation in the following sense

$$\begin{aligned} g_{\mathrm{K}} &= \bar{g}_{\mathrm{K}} n^4, \\ \frac{dn}{dt} &= \alpha_n(1-n) - \beta_n n, \end{aligned} \tag{2.2}$$

where $\bar{g}_{\mathrm{K}}$ has the dimension $\mathrm{mS/cm^2}$, α_n and β_n are constant in time but depend on membrane potential V and n is a dimensionless variable constrained between 0 and 1. In the resting state V = 0, n has the value

$$n_0 = \frac{\alpha_n|_{V=0}}{\alpha_n|_{V=0} + \beta_n|_{V=0}}. \tag{2.3}$$

In order to obtain an explicit formulation for α_n and β_n, many measuraments were taken at different temperatures, ionic concentrations and membrane potential configurations and then plotted against the resting potential V to give the following result

$$\begin{aligned} \alpha_n &= 0.01 \frac{V+10}{e^{\frac{V+10}{10}} - 1}, \\ \beta_n &= 0.125 e^{\frac{V}{80}}, \end{aligned}$$

where the functions α_n and β_n are measured in ms and V is measured in mV. They noticed that during the depolarisation phase the variation in g_{K} has a sigmmoid shape, while the polarisation phase behaves as a decaying exponential: this situations are captured very well by α_n and β_n respectively. The fourth power that relates n to g_{K} is related to the number of first-order processes that have to occur in order to have the desired curvature of the sigmoid function: the more numerous they are, the more steep the shape will be. This is compatible with their educated guess (verified several years

later) that each ionic channel contains several gates, and all of them have to be open in order to have an open channel (and so a sigmoid depolarisation) while it is sufficient that one of them is closed to shut the channel (hence the exponentially decaying polarisation). After this physical explanation, n can be interpreted as the probability for a given gate to be open: if we extend this to the more realistic case of a large number of channels, it becomes the fraction of the total number of gates that are in the permessive state (N.B. The definition of permissive/non-permissive state of the gate is a little bit different from the definition of open/close and it was given later than [12], but still today α and β represents the transition between this two states).
Regarding the sodium conductance, the procedure followed is only slightly different from the one adopted for the potassium. As before the idea is that the conductance is related to a certain number of first order process, but while for the potassium these processes are all of the same kind, described by the gate state n, now we have two different variables that come into play. This led them to the following assumptions:

$$g_{\mathrm{Na}} = m^3 h \bar{g}_{\mathrm{Na}},$$
$$\frac{dm}{dt} = \alpha_m(1-m) - \beta_m m, \tag{2.4}$$
$$\frac{dh}{dt} = \alpha_h(1-h) - \beta_h h, \tag{2.5}$$

where $\bar{g}_{\mathrm{Na}}$ is a constant and the α's and β's depend on V but not on time. Here h plays an opposite role of what n did for the potassium channels: in this case we consider it to be means when the channel is inactive and we have the polarisation phase. On the other hand, m behaves as n, with the only difference that for sodium ion channels the steepness of the relation between gate conductance and membrane potential is given by only 3 of these gates instead of 4 as before. However, even if all these "activating" gates m are open, but the "inactivating" gate h is open, then the channel remains closed. The initial states for the first-order equations describing the transition rates are again

$$m_0 = \frac{\alpha_m|_{V_0}}{\alpha_m|_{V_0} + \beta_m|_{V_0}}, \tag{2.6}$$
$$h_0 = \frac{\alpha_h|_{V_0}}{\alpha_h|_{V_0} + \beta_h|_{V_0}}. \tag{2.7}$$

Now that we have all the coefficients, we can rewrite (2.1) as

$$I = C_M \frac{\partial V}{\partial t} + \bar{g}_{\mathrm{K}} n^4 (V - V_{\mathrm{K}}) + \bar{g}_{\mathrm{Na}} m^3 h (V - V_{\mathrm{Na}}) + \bar{g}_{\mathrm{L}} (V - V_{\mathrm{L}}). \tag{2.8}$$

For the cable approximation of the axon, which is surrounded by a large amount of conducting fluid, the membrane current, in the case of a propa-

gated action potential, is related to the potential V by the following electrostatical relation

$$I = \frac{1}{2aR}\frac{\partial}{\partial x}\left(a^2\frac{\partial V}{\partial x}\right),$$

where R is the cytoplasmatic resistivity and a the axon radius, which may vary along the cable. Hence, after rearranging equation (2.8) to take into account the unit of measuraments used for the experimental setting, we obtain

$$\begin{aligned} 10^{-3}C_M\frac{\partial V}{\partial t} =& \frac{10^4}{2aR}\frac{\partial}{\partial x}\left(a^2\frac{\partial V}{\partial x}\right) \\ & - \bar{g}_{\mathrm{K}}n^4(V - V_{\mathrm{K}}) - \bar{g}_{\mathrm{Na}}m^3h(V - V_{\mathrm{Na}}) - \bar{g}_{\mathrm{L}}(V - V_{\mathrm{L}}) \\ & - \frac{10^2 g_{syn}y}{2\pi a}\delta_{syn}(x)(V - V_{syn}), \end{aligned} \tag{2.9}$$

where $\delta_{syn}(x)$ is used to represents the position of the synaptic receptor, in which we will start a spike after a time t_{off}. Here we consider the most simple modelization of synaptic interaction $g_{syn}(y)$, depending only on the synaptic receptor state y evolving as

$$y(t) = \frac{t - t_{\mathrm{off}}}{\tau}e^{\left(-\frac{t-t_{\mathrm{off}}-\tau}{\tau}\right)},$$

and τ is used as a timing parameter to adjust the duration of the spike. This is a simple, yet effective, description of an incoming spike in the potential, able to capture the nature of the synaptic behaviour, but also to allow fast and efficient computation, required in the case of large neural networks.

2.4 Problem statement

The model described in the previous section is applied on a domain which can be schematized as the one represented in Figure (2.4). For our simulation we chose to have the same length for Ω_A and Ω_{B_1} ($250\mu m$), while Ω_{B_2} is taken shorter ($150\mu m$).

The unbranched sections of the neuron are represented as 1D segments, an approximation due to the difference between length and thickness of each branch: this entails that all the variables considered are function of the axial abscissa along the cable but not of the radial distance from the center. This model is acceptable if the real radius, even if variable, show only relative deviations with respect to its mean value: some problem may occur if, for instance, we consider a tapered dendrite. This is not the case since the radius used is $a = 0.5\mu m$ for the entire length of each of the branches. It can be considered an approximation of the mean thichkness of a dendrite, but in more realistic simulations this parameter is strongly based on previously obtained experimental data.

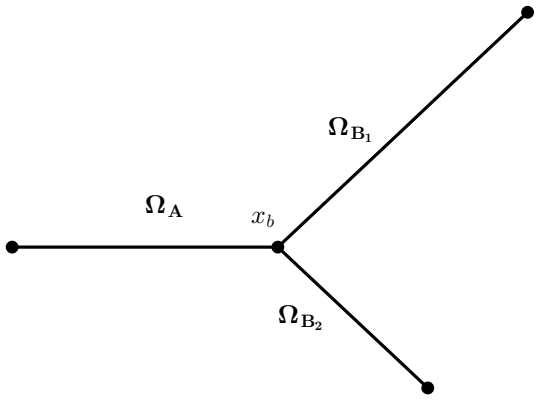

Figure 2.4: Dendritic tree with one branching node.

To set the boundary conditions and then complete the statement of the problem, electrostatic consideration are needed: the current between two points x and $x + \Delta x$ is proportional to the voltage difference and the inverse of membrane resistance, thus

$$I(x) = \frac{\pi a^2}{R} \frac{V(x - \Delta x) - V(x)}{\Delta x},$$

and by taking the limit $\Delta x \to 0$ we have

$$I(x) = \frac{\pi a^2}{R} \frac{\partial V}{\partial x}.$$

In the branching node, Kirchoff's law for the conservation of electric charge says that

$$\left(a^2 \frac{\partial V}{\partial x}\right)\Big|_{x=x_b}^{A} - \left(a^2 \frac{\partial V}{\partial x}\right)\Big|_{x=x_b}^{B_1} - \left(a^2 \frac{\partial V}{\partial x}\right)\Big|_{x=x_b}^{B_2} = 0,$$

which is chosen as boundary condition in the branching node x_b.
On the terminal points it is assumed that no current flows outside or inside, giving homogeneous Neumann conditions

$$\frac{\partial V}{\partial x} = 0.$$

To complete the problem defined by (2.9), (2.2), (2.4) and (2.5) a starting condition is needed, since we have a first derivative in time. Regarding the gate states n, m and h we use the one originally derived by Hodgkin and

Huxley (2.3), (2.6) and (2.7) while for the potential we start with a constant value, equal to $-64.974\ mV$. We expect an homogeneous scenario until the arrival of the spike, at $t_{\text{off}} = 1ms$

Chapter 3

Space discretization of PDEs system on a branched domain

Equations (2.9),(2.2),(2.4) and (2.5) represent a set of partial differential equations (PDEs)[19], equations that contain multivariate functions and their partial derivatives, for which an explicit analytical solution is not known. Hence a numerical approximation is needed. The method of lines (MOL)[21] is a technique that allows to obtain, from the system of PDEs, a system of ordinary differential equations (ODEs), which is assumed to be easier to analyze and solve. In this chapter we will explain more in detail this method applied to our problem and we will present the grid chosen for discretizing our domain, in a similar way to how is implemented in NEURON code .

3.1 The Method of Lines

The broad idea of MOL is to discretize all but one dimension. The most common setting in which this method is applied is the case of a problem with space and time as independent variables. The space is discretized first, leaving a continuos time-dependent set of problems, which can be solved directly or by using numerical approximations. This procedure allows to split the solution of the problem in different steps, and for each of them using general purpouse or problem dependent solver, both for space discretization and time integration.

A general class of problems, called parabolic problems, to which also the cable equation belongs, can be written as

$$\begin{cases} \frac{\partial}{\partial t}u(\mathbf{x},t) = \mathcal{L}u(\mathbf{x},t), \\ u(\mathbf{x},t_0) = u_0 \end{cases} \tag{3.1}$$

where $\mathbf{x}$ and t are space and time variables, u is the unknown solution, equal to u_0 at the time t_0 and $\mathcal{L}$ is a second order elliptic operator in space. An el-

liptic operator is characterized by not having characteristic directions, which could give discontinuities or shocks, as it happens for hyperbolic problems. As first step of MOL a suited approximation of the problem is given by a space discretization, leading to

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathcal{L}_h\mathbf{u}(t) \tag{3.2}$$

In (3.2) $\mathbf{u}$ represents the approximation of the solution: the i-th component u_i is related to the i-th node of the grid. The single PDE is replaced by a system of ODEs, one for each node used for the discretization of the domain(see Figure 3.1). It must be clear that the set of equations we are actually solving does not have the same solution of (3.1) in most of the cases. Depending on the discretization technique used we have

$$u(x_i, t) = u_i(t) + \mathcal{O}(h^p), \tag{3.3}$$

where p is the order of accuracy of the discretization scheme used. Regarding these schemes the most used are:

- *Finite Difference Method*: Algebraic approximations of derivatives and dominant approach for parabolic equations in the case of simple domains. In more complex cases, they are not easy to implement, since they are based on Taylor expansion. [17].

- *Finite Volume Method*: Starting from the integral form of (3.1) a "volume" around each point of the grid is defined and then, using the divergence theorem, a set of equations is derived from the conservations of quantities in the volumes, given a definition of flux between different volumes. These methods are suited for hyperbolic problems, since their conservative nature and they manage better complex geometries [15].

- *Finite Element Method*: Based on the variational formulation of the problem (the most common is Galerkin formulation), the broad idea of this technique is to split the original problem into smaller cases (finite elements), and then to assemble all the equation governing each element into a larger system. With this tecnique it is easier to deal with local issues like discontinuities or particularly complex shape of the domain [24].

In certain restricted cases these methods are equivalent. For instance for 1D problems, involving only reaction and diffusive terms, FDM are equivalent to FEM up to the second order if for the latter we use mass lumping approximation.

In short MOL is not a single and defined way of dealing with PDEs, but more a general approach that should be specified for each problem we are

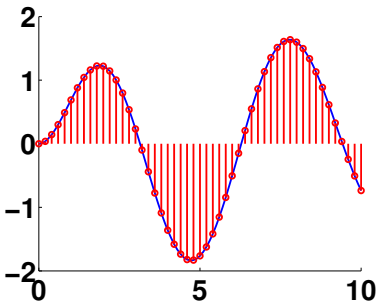

Figure 3.1: Example of MOL. The discretization in space leads to a set of problems depending on time. The reason for the term "lines" in the name of the method can be guessed from this figure: a juxtaposition of lines is obtained.

dealing with, given a preliminary study of possible discontinuities or instability issues. In the next section we will go more into depth with FDM, which is the method chosen for NEURON code, given its ease of implementation for the domain taken into consideration and the continuous nature of the solution.

3.2 Second Order Finite differences

According to the biological model [20], a multicompartimental approach is used, so that each node of the mesh represents the center of a compartiment, the unit element in which we split the dendrite. In our case we will deal with uniform compartiments, but this doesn't imply a uniform discretization since we have also to consider the boundary and branching nodes (see Figure 3.2). On the internal regions of the dendrite a uniform mesh with spacing Δx is used, while for the elements near the boundary is reduced to $\frac{\Delta x}{2}$.

The terms in our set of equations that will cause more troubles for the spatial discretization are the second order operator and the δ in the cable equation. We will explain in details the choice for the former and we give an idea of approach used for the latter in NEURON code.

3.2.1 Second order term

A possible starting point for the discretization of derivatives is to look at finite differences centered at each point x_j of the mesh: here j represents the position in the enumeration used in Figure 3.2, which is useful for implicit methods and will be explained in details in an apposite chapter later. The points far from boundaries the following standard results for uniform grid

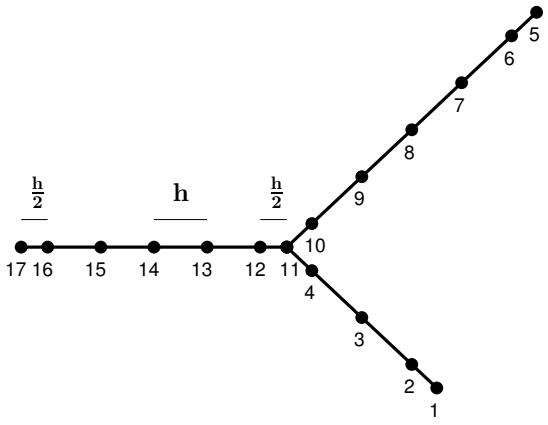

Figure 3.2: Example of mesh used where $h = 50\mu m$.

can be used

$$\frac{\partial f}{\partial x}(x)\bigg|_{x=x_j} = \frac{f(x_{j+\frac{1}{2}}) - f(x_{j-\frac{1}{2}})}{\Delta x} + \mathcal{O}(\Delta x^2), \tag{3.4}$$

which is a second order centered approximation of the first order derivative of a function where $x_{j+\frac{1}{2}} = x_j + \frac{\Delta x}{2}$ and $x_{j-\frac{1}{2}} = x_j - \frac{\Delta x}{2}$. By applying this form twice in the diffusive term of (2.9) we obtain

$$\frac{\partial}{\partial x}\left(a^2\frac{\partial V}{\partial x}\right)\bigg|_{x=x_j} = \frac{1}{\Delta x^2}(a^2_{j+\frac{1}{2}}(V_{j+1} - V_j) - a^2_{j-\frac{1}{2}}(V_j - V_{j-1})) + \mathcal{O}(\Delta x^2). \tag{3.5}$$

where $a_j = a(x_j)$ and $V_j = V(x_j)$. In the case of a constant (our setting), (3.5) can be reduced to

$$\frac{\partial}{\partial x}\left(a^2\frac{\partial V}{\partial x}\right)\bigg|_{x=x_j} = a^2\frac{V_{j+1} - 2V_j + V_{j-1}}{(\Delta x)^2} + \mathcal{O}(\Delta x^2).$$

Near the terminal and ending nodes the situation is more subtle because of the nonuniformity of the grid and so we need to go more in depth with Taylor series [23].
In following part of the section, as example, the case of the last node of the branch Ω_A is reported with the steps used for the derivation of the associated equation: the other cases follow exactly the same procedure, but for different indeces.

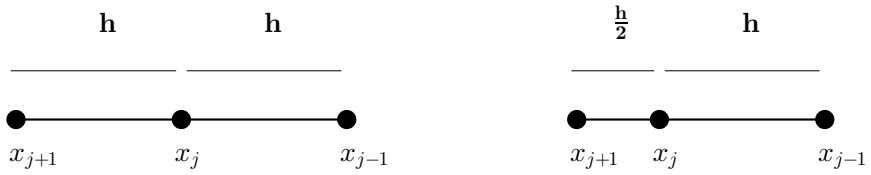

Figure 3.3: Local region in the middle of the mesh (left) and near the terminal point of the branch Ω_A (right)

Consider

$$f(x_{j+\frac{1}{2}}) = f(x_j) + \frac{\Delta x}{4}\frac{\partial f}{\partial x}(x)\bigg|_{x=x_j} + \left(\frac{\Delta x}{4}\right)^2 \frac{\partial^2 f}{\partial x^2}(x)\bigg|_{x=x_j} + \mathcal{O}(\Delta x^3),$$

$$f(x_{j-\frac{1}{2}}) = f(x_j) - \frac{\Delta x}{2}\frac{\partial f}{\partial x}(x)\bigg|_{x=x_j} + \left(\frac{\Delta x}{2}\right)^2 \frac{\partial^2 f}{\partial x^2}(x)\bigg|_{x=x_j} + \mathcal{O}(\Delta x^3),$$

and by combining them to kill the second order term we get

$$\frac{\partial f}{\partial x}(x)\bigg|_{x=x_j} = \frac{4f(x_{j+\frac{1}{2}}) - 3f(x_j) - f(x_{j-\frac{1}{2}})}{\frac{3}{2}\Delta x} + \mathcal{O}(\Delta x^2) \tag{3.6}$$

From a qualitative analysis, (3.6) makes sense: the coefficient related to the node towards the boundary ($x_{j+\frac{1}{2}}$) is more relevant than the one towards the middle of the branch ($x_{j-\frac{1}{2}}$), since the former is closer to the point for which we want an approximation of the solution (x_j). For the moment, with (3.6), we have only an approximation of the derivative of a general function $f(x)$. In our case we have

$$f(x) = a(x)^2 \frac{\partial V}{\partial x}(x)$$

and so to proceed we need approximations of $f(x_j)$, $f(x_{j+\frac{1}{2}})$ and $f(x_{j-\frac{1}{2}})$. For the last two we can use the previous result (3.4), since $x_{j+\frac{1}{2}}$ and $x_{j-\frac{1}{2}}$ are middle points of intervals $[x_{j+1}, x_j]$ and $[x_j, x_{j-1}]$ respectively:

$$f(x_{j+\frac{1}{2}}) = a^2_{j+\frac{1}{2}} \frac{\partial V}{\partial x}(x)\bigg|_{x_{j+\frac{1}{2}}} = a^2_{j+\frac{1}{2}} \frac{V_{j+1} - V_j}{\frac{\Delta x}{2}} + \mathcal{O}(\Delta x^2) \tag{3.7}$$

$$f(x_{j-\frac{1}{2}}) = a^2_{j-\frac{1}{2}} \frac{\partial V}{\partial x}(x)\bigg|_{x_{j-\frac{1}{2}}} = a^2_{j-\frac{1}{2}} \frac{V_j - V_{j-1}}{\Delta x} + \mathcal{O}(\Delta x^2) \tag{3.8}$$

Regarding $f(x_j)$ we need to approximate the term $\frac{\partial V}{\partial x}(x)$ on a nonuniform local grid. We do it by (3.6), replacing Δx with $\frac{\Delta x}{2}$, obtaining

$$f(x_j) = a_j^2 \frac{\partial V}{\partial x}(x)\bigg|_{x=x_j} = a_j^2 \frac{4V_{j+1} - 3V_j - V_{j-1}}{3\Delta x} + \mathcal{O}(\Delta x^2) \tag{3.9}$$

By replacing (3.7),(3.8) and (3.9) in (3.6) and reporting only the a constant case (for the sake of representation) we have

$$\frac{\partial}{\partial x}\left(a^2 \frac{\partial V}{\partial x}\right)\bigg|_{x=x_j} = a^2 \frac{4V_{j+1} - 6V_j + 2V_{j-1}}{\frac{3}{2}\Delta x^2} + \mathcal{O}(\Delta x^2) \tag{3.10}$$

3.2.2 Delta term and rest of the system

Regarding the δ term in (2.9) we give only the idea behind its approximation, relaxing the distribution theory used more rigorously by BBP team for their simulation. We exploit the equivalence between FD and FEM in our setting, starting from the latter to derive a result for the former. In FEM theory, the approximation of the solution is searched in a particular functional space, usually $\mathcal{H}^1$. A set of functions $\{\phi_i\}$ is chosen as base for the functional space and then the idea is to find the function $u_h \in \mathcal{H}^1$ which minimizes the distance to the solution of our problem. Hence, the unknowns in the FEM formulation are the coefficients of u_h with respect to $\{\phi_i\}$. The only term which contributes to the coefficient related to the δ can be found by computing (see [19] for more details)

$$\int_{x_{i-1}}^{x_{i+1}} \delta(x - x_j)\phi_i^2(x)dx = \int_{\mathbb{R}} \delta(x - x_j)\phi_i^2(x)dx = \phi_i^2(x_j) =$$
$$= \begin{cases} 1, & j = 1 \\ 0, & \text{otherwise} \end{cases} = \delta_{ij} \tag{3.11}$$

since ϕ_i is a piece-wise linear function which assumes value 0 in every node but the i-th, in which is equal to 1, and its support is $[x_{i-1}, x_{i+1}]$. In order to obtain the FD formulation of the δ we use the *mass lumping* technique, which consist in approximating the integral using a quadrature formula, in our case the trapezoidal rule

$$\int_{x_{i-1}}^{x_{i+1}} \delta(x - x_j)\phi_i^2(x)dx = 2\int_{x_{i-1}}^{x_i} \delta(x - x_j)\phi_i^2(x)dx \approx$$
$$\approx 2\frac{\Delta x}{2}\{\delta(x - x_j)\big|_{x=x_i}\phi_i^2(x_i) + \delta(x - x_j)\big|_{x=x_{i+1}}\phi_i^2(x_{i+1})\} =$$
$$= h\delta(x - x_j)\big|_{x=x_i} \tag{3.12}$$

By combining (3.12) and (3.11) we have

$$\delta(x - x_j)\big|_{x=x_i} \approx \frac{\delta_{ij}}{\Delta x} \tag{3.13}$$

Now that we managed the most troublesome cases for the cable equation, the rest of the system is easier to deal with. Since we don't have any derivatives, for a generic function $f(x)$ we have $f(x_j) \approx f_j$. Following the discretization of the rest of the system:

- *Cable equation:*
  - *Time derivative:*

$$10^{-3}\mathrm{c_m}\frac{\partial}{\partial t}V(x,t)\bigg|_{x=x_j} \approx 10^{-3}\mathrm{c_m}\frac{\mathrm{d}}{\mathrm{d}t}V_j(t)$$

  - *Ionic current:*

$$\begin{aligned}&-\mathrm{g_K}n^4(x,t)(V(x,t)-\mathrm{e_K}) - \mathrm{g_{Na}}m^3(x,t)h(x,t)(V(x,t)-\mathrm{e_{Na}}) - \mathrm{g_l}(V(x,t)-\mathrm{e_l})\bigg|_{x=x_j}\\ &\approx -\mathrm{g_K}n_j^4(t)(V_j(t)-\mathrm{e_K}) - \mathrm{g_{Na}}m_j^3(t)h_j(t)(V_j(t)-\mathrm{e_{Na}}) - \mathrm{g_l}(V_j(t)-\mathrm{e_l})\end{aligned}$$

- *Ion channels:*

$$\begin{aligned}&\frac{\partial}{\partial t}n(x,t) = \alpha_n(V)(1-n(x,t)) - \beta_n(V)n(x,t)\bigg|_{x=x_j}\\ &\Rightarrow \frac{\partial}{\partial t}n_j(t) = \alpha_n(V_j)(1-n_j(t)) - \beta_n(V_j)n_j(t)\end{aligned}$$

$$\begin{aligned}&\frac{\partial}{\partial t}m(x,t) = \alpha_m(V)(1-m(x,t)) - \beta_m(V)m(x,t)\bigg|_{x=x_j}\\ &\Rightarrow \frac{\partial}{\partial t}m_j(t) = \alpha_m(V_j)(1-m_j(t)) - \beta_m(V_j)m_j(t)\end{aligned}$$

$$\begin{aligned}&\frac{\partial}{\partial t}h(x,t) = \alpha_h(V)(1-h(x,t)) - \beta_h(V)h(x,t)\bigg|_{x=x_j}\\ &\Rightarrow \frac{\partial}{\partial t}h_j(t) = \alpha_h(V_j)(1-h_j(t)) - \beta_h(V_j)h_j(t)\end{aligned}$$

3.3 Boundary conditions

Since our boundary conditions, both for the terminal and branching nodes, represent constrains on the behaviour of fluxes, then spatial derivatives are involved. In order to discretize them, we first have to keep in mind possible performance drawbacks in the following computer implementation. As we can see in the approximation of the second order operator (3.10), for each

points we need also the value of the potential in its closest neighbours, giving a general tridiagonal pattern. This "pure" tridiagonal pattern is actually broken because of the presence of the branching node: whatever enumeration order or discretization we use, in the branching node neighbour set there will be always a point "far" from the branching node in terms of enumeration (see Figure 3.2). However, thanks to an algorithm presented in [11] and described more in detail later in this report, the efficiency of Thomas' solver for tridiagonal system solution can be recovered, drastically reducing the computational cost for implicit time integrators.

The properties that we want to conserve with the discretization of boundary conditions are:

1. Almost tridiagonal pattern
2. 2^{nd} order of accuracy in space

The complication here is that 2 could be obtained using a decentered 2^{nd} order scheme, at the cost of breaking 1, or by using a centered 2^{nd} order scheme, but in this case we will need informations outside the domain since we are working on boundaries. A possible (yet complex) way to manage the second case is to use the *ghost points* technique [5], but for our problem, a way simpler solution can be found. In some cases, as explained in [14], the global second order of convergence could be mantained even if we have isolated points dicretized with a first order approximation: the key for this results is the finer mesh used near the boundaries, which allow to use the simplest discretization and at the same time to achieve both 1 and 2

- Neumann conditions

$$\frac{\partial}{\partial x}$$

-

Chapter 4

Time integration

4.1 Stability of numerical integrators

4.1.1 Examples

4.1.2 Linear stability analysis

The more general case we can face is the non-linear non-automous ODEs

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{y} = \mathbf{f}(t, \mathbf{y}), & \mathbf{y} \in \mathbb{R}^n, t \in [t_1, t_2] \\ \mathbf{y}(t_0) = \mathbf{y}_0. \end{cases} \tag{4.1}$$

To study its stability behaviour we need to approximate this form to a case which is more manageable, by means of Taylor's series applied to the force term in (4.1). Suppose we have a smooth solution $\boldsymbol{\phi}(t)$, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{y} = \mathbf{f}(t, \boldsymbol{\phi}(t)) + \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(t, \boldsymbol{\phi}(t))(\mathbf{y}(t) - \boldsymbol{\phi}(t)) + \dots, \tag{4.2}$$

and then, by discarding all the terms of order bigger than 1 and setting $\mathbf{w}(t) = \mathbf{y}(t) - \boldsymbol{\phi}(t)$ in (4.2), we remain with

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{w} = \mathbf{A}(t)\mathbf{w}(t), \tag{4.3}$$

where $\mathbf{A}(t) = \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(t, \boldsymbol{\phi}(t))$. Now we have to kill the time dependency in $\mathbf{A}(t)$, by freezing $t = t_0$, in order to obtain a constant matrix $\mathbf{A}_0$. By doing this, the following analysis will be valid only near the starting time and to extend it further we have to change the istant in which the time is frozen. For the sake of clarity now we suppose the matrix A_0 to be fully diagonalizable, i.e. there exists an orthogonal matrix $\mathbf{P}$ s.t.

$$\mathbf{P}^{-1}\mathbf{A}_0\mathbf{P} = \mathbf{D}, \qquad D = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}. \tag{4.4}$$

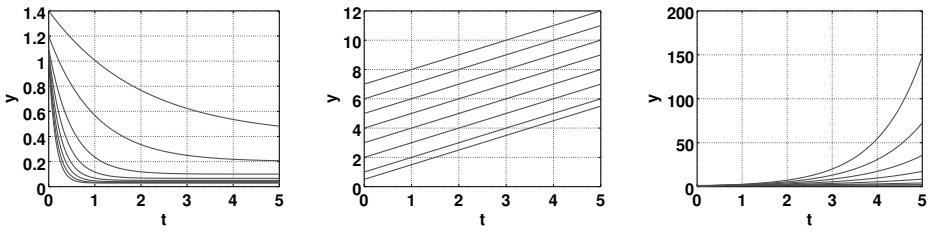

Figure 4.1: Solution of Dahlquist's equation for Re(λ)< 0 (stable), Re(λ)= 0 (asymptotically stable) and Re(λ)> 0 (unstable)

This is not always the case, but even in more complex scenarios where the diagonalization is not possible, the stability can be investigated requiring different conditions to A. Using (4.4) in (4.3) we are left with a set of n decoupled equations

$$\frac{\mathrm{d}}{\mathrm{d}t}v_i = \lambda_i v_i, \qquad i = 1 \dots n, \tag{4.5}$$

where λ_i-s are the eigenvalues of A_0. Each of them represents an example of Dalqhuist test equation

$$\frac{\mathrm{d}}{\mathrm{d}t}y = \lambda y, \qquad \lambda \in \mathbb{C}. \tag{4.6}$$

The solution for this famous problem is an exponential which slope depends on the sign of λ (Figure 4.1). Hence, the meaning of (4.5) is that even if only one of the eigenvalues of the linearized system has a real part greater than zero, then the solution is unstable [6]. A possible definition of stability was given by Lyapunov in [16]

Definition 1. A solution to (4.1) that exists for all $t \geq t_0$ is said to be stable (asyntotically stable) in the sense of Lyapunov if $\forall \epsilon > 0$ $\exists \delta > 0$ s.t. if $\|\Delta \mathbf{y}\| < \delta$ then

$$\left(\lim_{t=\infty}\right) \quad \|\mathbf{y}(t, t_0, \mathbf{y}_0) - \mathbf{y}(t, t_0, \mathbf{y}_0 + \Delta \mathbf{y})\| < \epsilon.$$

4.1.3 Stability of Runge–Kutta methods

The Runge–Kutta (RK) methods are the most common way to solve an ODE: they are a family of explicit and implicit iterative numerical schemes. Another important group of ODEs solver is the Linear multistep methods [10], which we will not deal with in this report. RK methods are based on the idea of computing the numerical solution after a time step Δt through a weighted sum of intermediate steps, which approximate the slope of the solution at specific points $\in [t, t + \Delta t]$. Any Runge Kutta method can be presented in the following general form

Definition 2. Let $s \in \mathbb{N}^+$ and let b_i, a_{ij} and c_i, $1 \leq i, j \leq s$ be real numbers. A s-stage Runge-Kutta method for the integration of

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t} y = f(t, y), \\ y(t_0) = y_0, \end{cases} \tag{4.7}$$

is given by

$$\mathrm{K}_i = f\left(t_0 + \mathrm{c}_i \Delta t, y_0 + \Delta t \sum_{j=1}^{s} \mathrm{a}_{ij} \mathrm{K}_j \right),$$
$$y_1 = y_0 + \Delta t \sum_{i=1}^{s} \mathrm{b}_i \mathrm{K}_i.$$

A RK method applied to (4.7) gives approximations at different times $\{y_n(\Delta t, \lambda)\}_{n \geq 0}$. Also for the numerical solution is possible to define a stability domain based on bounded values.

Definition 3. The stability domain of a RK method is

$$S_n = \{ z = \Delta t \lambda \in \mathbb{C} : \{y_n(\Delta t, \lambda)\}_{n \geq 0} \quad \text{remains bounded} \}$$

Usually this definition is reinterpreted in terms of stability function, which is introduced in the following statement

Definition 4. Following the same notation of Definition 2, using a RK method to solve Dahlquist's test problem, we obtain after one step $y_1 = R(\Delta t \lambda) y_0$, where

$$R(z) = 1 + \mathbf{b}^T z (I - z\mathbb{A})^{-1} \mathbb{I}, \qquad z = \Delta t \lambda$$
$$\mathbf{b} = \begin{pmatrix} \mathrm{b}_1 \\ \vdots \\ \vdots \\ \mathrm{b}_s \end{pmatrix}, \quad \mathbb{A} = \begin{pmatrix} \mathrm{a}_{1,1} & \mathrm{a}_{1,2} & \dots & \\ \mathrm{a}_{2,1} & \ddots & & \vdots \\ & & \ddots & \mathrm{a}_{n-1,n} \\ & \dots & \mathrm{a}_{n,n-1} & \mathrm{a}_{n,n} \end{pmatrix} \quad \mathbb{I} = \begin{pmatrix} 1 \\ \vdots \\ \vdots \\ 1 \end{pmatrix}$$

is called stability function.

This function does not identify a scheme, since two different numerical methods can have the same stability function. Since we want our method to be stable, the solution of the test problem need to be analyzed. Using Definition 4 we have

$$y_n = R(\Delta t \lambda) y_{n-1} = \dots = (R(\Delta t \lambda))^n y_0 < \infty$$

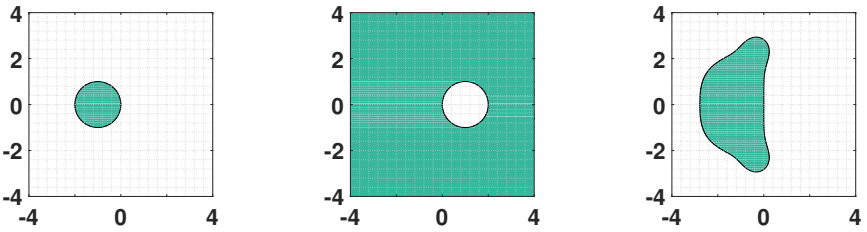

Figure 4.2: Stability regions of explicit Euler, implicit Euler and classical Runge-Kutta

for every y_0 if and only if $|R(\Delta t\lambda)| < 1$, which is an equivalent characterization of the stability domain.

As we have seen before, the test equation has a stable solution only if $\lambda \in \mathbb{C}^-$. A good property is that the numerical methods conserve the stability in the same case.

Definition 5. (Dahlquist 1963) A method, whose stability domain satisfies $S_n \in \mathbb{C}^-$ is called *A-stable*.

The stability function of a RK method can be expressed as the ratio between two polynomials of degree $\leq s$ for the implicit case and as a polynomial of the degree $\leq s$ for explicit method. Hence, by using the maximum principle theorem, it is possible to give sufficient and necessary conditions for the A-stability of a scheme.

Definition 6. A RK method is A-stable if and only if its stability function satisfies:

- $|\mathrm{R}(iy)| \leq 1 \quad \forall y \in \mathbb{R}$ (also called *I-stability* condition)
- $\mathrm{R}(z)$ is analitic in $\mathbb{C}^-$

The *A-stability* property ensures that our numerical solution does not explode, but there still could be some bounded oscillations. A stronger requirement is sufficient to guarante a better qualitative representation of the exact solution.

Definition 7. A RK method is *L-stable* if it is *A-stable* and $\lim_{z\to\infty} \mathrm{R}(z) = 0$

The limit for $z \to \infty$ can be replaced by $z \to -\infty$, since we are dealing with rational function, so we can focus on $\mathbb{C}^-$.

As can be seen in Figure 4.3, an *L-stable* implicit method could behave better than an only *A-stable* implicit method like the implicit midpoint, even if the former has a lower order of accuracy than the latter. This is due to the additional property $\mathrm{R}(\Delta t\lambda) \approx 0$ when we have big time steps ($\Delta t \gg \lambda^{-1}$). Since we are working with rational functions

$$\lim_{z\to-\infty} R(z) = \lim_{z=iy, y\to\infty} R(z), \tag{4.8}$$

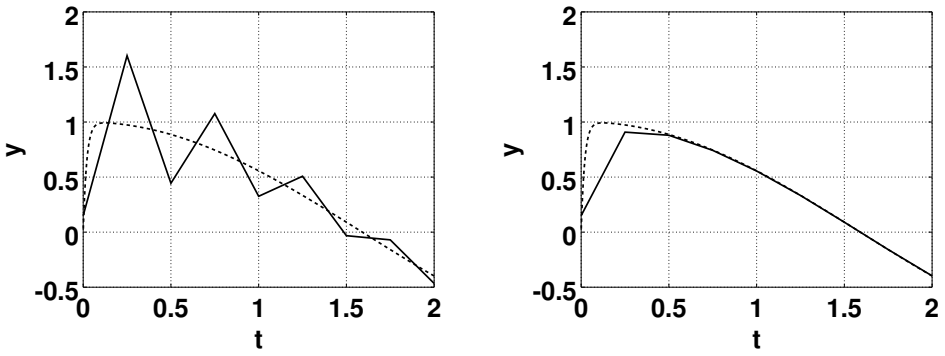

Figure 4.3: Solution of Curtis-Hirschfelder equation using implicit midpoint(left) and Euler implicit(right) methods. The former method is only *A-stable* while the latter is also *L-stable*. The dotted line represents in both cases the exact solution.

If the stability domain of the numerical method contains only $\mathbb{C}^-$ then $|R(iy)| = 1 \ \forall y \in \mathbb{R}$. But from (4.8) we have that, in case of z with a large negative part, $R(z)$ is almost equal to 1 (but smaller). Considering the role of $R(z)$ in Definition 4, the obscillations are only slghtly damped.

4.1.4 Stiff problems

The concern about stability of a numerical solution is due to the resolution of *stiff* problems. There is not a universally accepted definition of *stiffness* of an ODE problem, but a possible tentative characterization is the following

Definition 8. A stable problem $\frac{\mathrm{d}}{\mathrm{d}t}y = f(t, y)$ is called *stiff* when $\max_i |\lambda_i|$ is large, where λ_i is an eigenvalue of $\frac{\partial f}{\partial y}$, with coefficients frozen at some (t_0, y_0). $L := \max_i |\Re(\lambda_i)|$ is sometimes called *stiffness* index and gives an hint about the magnitude of the *stiffness*, while usually the letter ρ is usually used to refere to $\max_i |\lambda_i|$.

For a numerical method to be stable $z = \Delta t \rho$ need to be included in its stability domain. Since ρ could be even of the scale of 10^6 for common problems, a particularly small time step must be chosen in order to not have a solution that literally explodes after few steps. The time step required is influenced by different parameters, firstly by the nature of the problem, but also by the stability domain of the scheme employed. If the stablity domain is restricted to a small portion of the complex plane near the origin, a very fine time step is used, even if we are not interested in having a so detailed picture of the solution in the that time interval. This issue should be taken into account every time we have to chose an ODE solver: sometimes the benefits of using an easy-to-implement and fast method are vanished by the insanely high number of steps required to "calm down" the solution.

4.2 Implicit methods: Solving a linear system

4.2.1 Implicit methods properties

For implicit method, to find the stages defined in Definition 2 a system of non-linear equations has to be solved in order to find the different K_i. For fully implicit methods the dimensions of the system is equal to the number of stages, while for some other methods part of the stages could be determined explicitely. Even worse, if we are the dealing with an n-dimensional problem, the dimension grow to $n \times$ (number o implicit stages). This makes implicit methods more costly to implement and to be solved and more subject to round-off errors. In [10] some tricks to reduce this kind of errors and make the non-linear system easier to be solved iteratively are introduced: the authors also present good choices for the initial state and stopping criterion of the iterative procedure. One of the advices we want to report is the drawback of fixed point procedure to solve a non-linear system. Given a system of equation equation $\mathbf{x} = \mathbf{G}(\mathbf{x})$, similar to the one that we have to solve for the stages, and provided a reasonably good starting guess for the solution $\mathbf{x}^0$, this procedure requires to compute $\mathbf{x}^{k+1} = \mathbf{G}(\mathbf{x}^k)$ until the difference between two iterations satisfies the requested tolerance. To converge, we need to work with a contraction, i.e.

$$\|\mathbf{G}(\mathbf{x}) - \mathbf{G}(\mathbf{y})\| \leq \nu\|\mathbf{x} - \mathbf{y}\|, \qquad \nu < 1.$$

If we consider that

$$\mathbf{G}(\mathbf{x}) = \mathbf{G}(\mathbf{y}) + \mathbf{G}'(\mathbf{y})(\mathbf{x} - \mathbf{y}) + \dots,$$

and for the problem arising from implicit methods we have, if $\mathbf{x}^0$ is close to the solution then

$$\nu \sim \|\mathbf{G}'(\mathbf{y})\| \sim C\Delta t\rho.$$

But for stiff problems we know that ρ could be very large: this means that what we have gain by using an implicit method will be lost since a very fine time step is required for convergence. A better alternative to solve kind of problems is the Netwon-Raphson method, which relies on the computation

All these complexities are the price for a usually large stability domain. This is the reason for which they are the standard for solving really stiff problems. We must pay attention to not think that any time discrretization will give a stable numerical solution: while it is true that the stability region is bigger than the one of explicit methods, only a sub set of implicit methods are *A-stable* (see Figure 4.4).

4.2.2 Strang splitting

As we have seen in the previous section the non-linearity of the problem could cause some issues regarding instabilty and difficulties in the implementation.

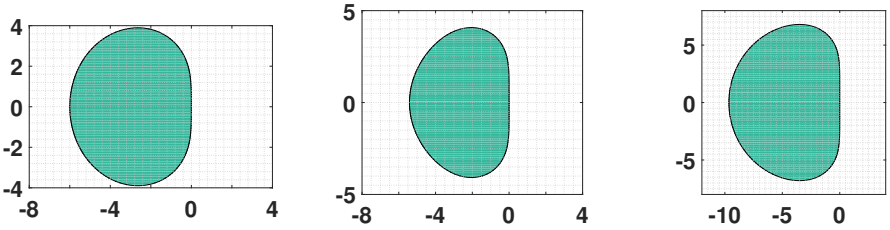

Figure 4.4: Stability regions of explicit Euler, implicit Euler and classical Runge-Kutta

For particular class of operators the non-linearity could be killed by splitting the problem in the sum of linear operators, much easier to deal with. Here we will present a possible technique to address this splitting, with the approach *divide et impera*.

Suppose we have the following differential equation that we want to integrate for a single time step Δt

$$\frac{\mathrm{d}}{\mathrm{d}t}y = f(y,t) = f_1(y,t) + f_2(y,t), \tag{4.9}$$

and that

$$\frac{\mathrm{d}}{\mathrm{d}t}y = f_1(y,t),$$
$$\frac{\mathrm{d}}{\mathrm{d}t}y = f_2(y,t),$$

happen to be easier problems than the original (no non-linearities, know exact solutions, ...). The idea for the construction of the new method is to solve first one of the two problems with a certain initial data

$$\frac{\mathrm{d}}{\mathrm{d}t}y = f_1(y,t), \qquad y(0) = y_n.$$

This take us to $y^* = y_1(\Delta t; y_s)$, where y_1 represents the solution of the differential equation on vector field f_1. As next step, we start from y^* and solve the second initial value problem

$$\frac{\mathrm{d}}{\mathrm{d}t}y = f_2(y,t), \qquad y(0) = y^*.$$

on the same interval Δt to obtain $y_{n+1} = y_2(\Delta t; y^*)$ It is possible to reinterpret this procedure in terms of *flows*[10]

Definition 9. The *flow* of a system $\frac{\mathrm{d}}{\mathrm{d}t} = f(y,t)$ is the mapping which, to any point y_0 in the phase space, associates the value $y(t)$ of the solution with initial value $y(0) = y_0$. This map, denoted by ϕ_t, is thus defined by

$$\phi_t(y_0) = y(t), \quad if \quad y(0) = y_0.$$

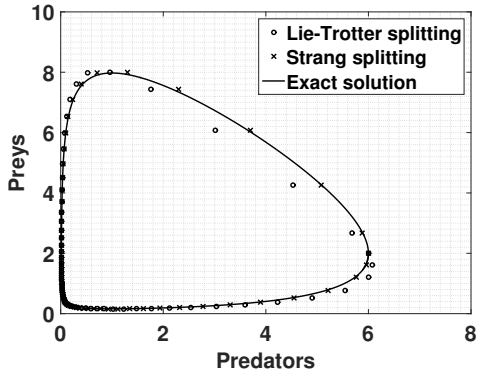

Figure 4.5: Solution of Loitemtka-Volterra equation [**Lotka**]. Comparison between exact solution and different splitting techniques.

Denoting the flow related to f_1 and f_2 as ϕ_{t,f_1} and ϕ_{t,f_2} respectively, what we have previously computed is $y_{n+1} = \phi_{\Delta t,f_2}(\phi_{\Delta t,f_1}(y_n)) = (\phi_{\Delta t,f_2} \circ \phi_{\Delta t,f_1})(y_n) = \phi^{T-L}_{\Delta t,f}(y_n)$. This is an example of a *splitting method*, the *Lie-Trotter splitting*. Since the order of the operators is arbitrary, also $y_{n+1} = (\phi_{\Delta t,f_1} \circ \phi_{\Delta t,f_2})(y_n) = (\phi^{T-L}_{\Delta t,f})^*(y_n)$ would have been a viable choice (and we obtain the adjoint of the previous operator). By Taylor's expansion we can prove that

$$(\phi_{\Delta t,f_1} \circ \phi_{\Delta t,f_2})(y_0) = \phi_{\Delta t,f}(y_0) + \mathcal{O}(h^2)$$
$$(\phi_{\Delta t,f_2} \circ \phi_{\Delta t,f_1})(y_0) = \phi_{\Delta t,f}(y_0) + \mathcal{O}(h^2)$$

and so both the methods gives only a 1^{st} order approximation of the solution to (4.9)

4.3 Explicit methods: Stabilizing with stages

Explicit stabilized RK methods are explicit methods with a stability domain extended along the real negative axis until a required value. They are suited for large system of ODEs, usually originating from the space discretization of a parabolic problem, like the case we are dealing with. They are efficient tools, since they are able to manage stiff problems by preserving the most important advantage of explicit methods, since they do not require to solve a linear system each step. This advantage comes with the price of using a larger number of stages for each step with respect to traditional explicit RK methods of the same order of accuracy. The main idea is that a larger number of stages is required for increasing the size of the stability domain, instead of increasing the accuracy of the method. This approach, first introduced in [7], was then exploited to build the method called RKC, presented in [22]. Here we will study, implement and analyze the result of a successive

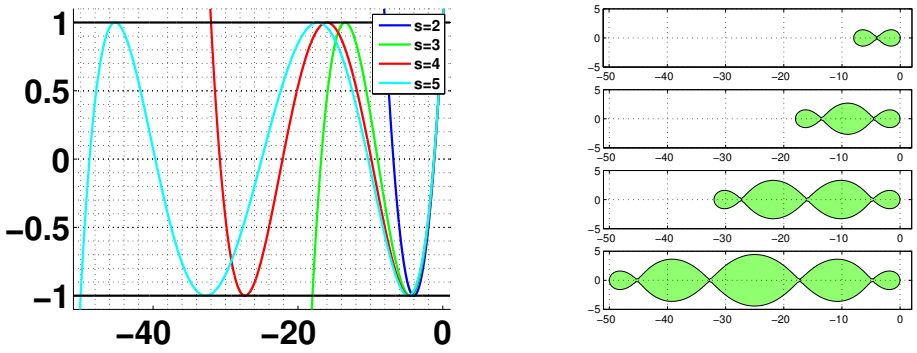

Figure 4.6: Shifted Chebyshev polynomials on the left (for $s = 2, 5$ and 10) and related stability domain on the right(for $s = 2, 5$ and 10, from top to bottom)

method, called ROCK2 (see [1] for more details). For almost all this kind of method there are two fundamental steps to be followed for the derivation of a numerical scheme:

1. Define a stability polynomial with certain property.
2. Construct a numerical method characterized by the stability function previously defined.

In the case of first order of accuracy, the optimal stability polynomials are the shifted Chebyshev polynomials $R_s = T_s(1 + \frac{z}{s^2})$, with s number of stages and $z \in \mathbb{C}$ (see figure 4.6). They can defined recoursively by the following formula

$$\begin{cases} R_0(z) = 1, \quad R_1(z) = z, \\ R_j(z) = 2zR_{j-1}(z) - R_{j-2}, \quad 2 \leq j \leq s. \end{cases}$$

They are first order consistent, which means that

$$R_s(z) = 1 + z + \mathcal{O}(z^2),$$

and they are limited between -1 and 1.

Chapter 5

Setting of the problem

Cable equation:

$$\begin{aligned} 10^{-3} C_M \frac{\partial V}{\partial t} =& \frac{10^4}{2aR} \frac{\partial}{\partial x} \left(a^2 \frac{\partial V}{\partial x} \right) \\ & - \bar{g}_{\mathrm{K}} n^4 (V - V_{\mathrm{K}}) - \bar{g}_{\mathrm{Na}} m^3 h (V - V_{\mathrm{Na}}) - \bar{g}_{\mathrm{L}} (V - V_{\mathrm{L}}) \\ & - \frac{10^2 g_{syn} y}{2\pi a} \delta_{syn}(x)(V - V_{syn}), \end{aligned}$$

Boundary conditions:

- Branching node

$$\left(a^2 \frac{\partial V}{\partial x} \right) \Big|_{x=x_b}^{A} = \left(a^2 \frac{\partial V}{\partial x} \right) \Big|_{x=x_b}^{B_1} + \left(a^2 \frac{\partial V}{\partial x} \right) \Big|_{x=x_b}^{B_2}$$

- Terminal nodes

$$\frac{\partial V}{\partial x} = 0$$

Initial condition:

$$V \Big|_{t=0} = V_0$$

Gate states:

$$\begin{aligned} \frac{dn}{dt} &= \alpha_n (1 - n) - \beta_n n \\ \frac{dm}{dt} &= \alpha_m (1 - m) - \beta_m m \\ \frac{dh}{dt} &= \alpha_h (1 - h) - \beta_h h \end{aligned}$$

Initial conditions

$$n_0 = \frac{\alpha_n|_{V=0}}{\alpha_n|_{V=0} + \beta_n|_{V=0}}$$
$$m_0 = \frac{\alpha_m|_{V=0}}{\alpha_m|_{V=0} + \beta_m|_{V=0}}$$
$$h_0 = \frac{\alpha_h|_{V=0}}{\alpha_h|_{V=0} + \beta_h|_{V=0}}$$

Bibliography

[1] Assyr Abdulle and Alexei A Medovikov. "Second order Chebyshev methods based on orthogonal polynomials". In: *Numerische Mathematik* 90.1 (2001), pp. 1–18.

[2] Therese Abrahamsson et al. "Thin dendrites of cerebellar interneurons confer sublinear synaptic integration and a gradient of short-term plasticity". In: *Neuron* 73.6 (2012), pp. 1159–1172.

[3] Merrill J Birdno et al. "Stimulus features underlying reduced tremor suppression with temporally patterned deep brain stimulation". In: *Journal of neurophysiology* 107.1 (2012), pp. 364–383.

[4] Nicholas T Carnevale and Michael L Hines. *The NEURON book*. Cambridge University Press, 2006.

[5] Armando Coco and Giovanni Russo. "Finite-difference ghost-point multigrid methods on Cartesian grids for elliptic problems in arbitrary domains". In: *Journal of Computational Physics* 241 (2013), pp. 464–501.

[6] Germund G Dahlquist. "A special stability problem for linear multistep methods". In: *BIT Numerical Mathematics* 3.1 (1963), pp. 27–43.

[7] Pieter J van Der Houwen and Ben P Sommeijer. "On the Internal Stability of Explicit, m-Stage Runge-Kutta Methods for Large m-Values". In: *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik* 60.10 (1980), pp. 479–485.

[8] André A Fenton et al. "Attention-like modulation of hippocampus place cell discharge". In: *The Journal of Neuroscience* 30.13 (2010), pp. 4613–4625.

[9] EG Gray. "The Fine Structure of the Nervous System". In: *Journal of anatomy* 108.Pt 1 (1971), p. 198.

[10] E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2008. ISBN: 9783540566700. URL: `https://books.google.ch/books?id=F93u7VcSRyYC`.

[11] Michael Hines. "Efficient computation of branched nerve equations". In: *International journal of bio-medical computing* 15.1 (1984), pp. 69–76.

[12] AL Hodgkin and AF Huxley. "Propagation of electrical signals along giant nerve fibres". In: *Proceedings of the Royal Society of London. Series B, Biological Sciences* (1952), pp. 177–183.

[13] Eric R Kandel, James H Schwartz, Thomas M Jessell, et al. *Principles of neural science*. Vol. 4. McGraw-hill New York, 2000.

[14] Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. Vol. 98. Siam, 2007.

[15] Randall J LeVeque. *Finite volume methods for hyperbolic problems*. Vol. 31. Cambridge university press, 2002.

[16] Aleksandr Mikhailovich Lyapunov. "The general problem of the stability of motion". In: *International Journal of Control* 55.3 (1992), pp. 531–534.

[17] Andrew Ronald Mitchell and David Francis Griffiths. *The finite difference method in partial differential equations*. John Wiley, 1980.

[18] Don Monroe. "Neuromorphic computing gets ready for the (really) big time". In: *Communications of the ACM* 57.6 (2014), pp. 13–15.

[19] Alfio Quarteroni. *Numerical models for differential problems*. Vol. 2. Springer Science & Business Media, 2010.

[20] Wilfrid Rall. "Theoretical significance of dendritic trees for neuronal input-output relations". In: *Neural theory and modeling* (1964), pp. 73–97.

[21] William E Schiesser. *The numerical method of lines: integration of partial differential equations*. Elsevier, 2012.

[22] BP Sommeijer, LF Shampine, and JG Verwer. "RKC: An explicit solver for parabolic PDEs". In: *Journal of Computational and Applied Mathematics* 88.2 (1998), pp. 315–326.

[23] DB Spalding. "A novel finite difference formulation for differential expressions involving both first and second derivatives". In: *International Journal for Numerical Methods in Engineering* 4.4 (1972), pp. 551–559.

[24] Gilbert Strang and George J Fix. *An analysis of the finite element method*. Vol. 212. Prentice-Hall Englewood Cliffs, NJ, 1973.