POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Mathematical Modelling of Neurodegenerative Disorders

Semester Project
Numerical Analysis for Partial Differential Equations

Pettenon Francesco, Rinaldoni Davide, Venturi Francesca,

Abstract: The study presents a mathematical model for the understanding of neurodegenerative disorders, specifically Alzheimer's Disease, which are characterized by the spread of misfolded proteins through the brain tissue. These proteins are able to propagate their misfolded conformation to other proteins, leading to the accumulation of harmful aggregates in the brain and ultimately the degeneration of brain cells. The aim of the model is to better understand the dynamics of proteins' spreading and how it contributes to the progression of the disorder. To model the spread of proteins, the Fisher-Kolmogorov equation is used, which describes the diffusion of proteins through the brain tissue. Since the domain is not trivial at all, finite elements approximations may become computationally heavy, which is why brain network models are employed to make the model more tractable. Clinical images, specifically Magnetic Resonance Imaging scans, carry medical information: they are processed by means of the program DSI studio, that is able to create graphs and extract the Laplacian matrix as a representation of the diffusion tensor. The Laplacian graph captures the structural connectivity of the brain, which is important for a better understanding of the spreading of proteins through the tissue. Once the brain network has been established, the spread of the proteins is modeled through a system of ordinary differential equations, with various numerical methods applied to solve the equations. Simulations are run on patients of different ages and health states, with initial concentrations of misfolded proteins in a specific brain region. The results of the simulations show that the proteins spread to all nodes within a time frame of 10-50 years, depending on the value of the conversion parameter in the model. This time frame is consistent with the observed progression of neurodegenerative disorders in patients. In addition to the simulations, the predictions of the model are compared to real-case diffusion data to assess the accuracy of the model. The results show that the model is consistent with the observed protein spread in certain brain regions, providing evidence for the usefulness of the study in understanding the dynamics of neurodegenerative disorders. Overall, the model offers a new approach to studying the spread of misfolded proteins and the progression of neurodegenerative disorders. This study may eventually lead to new insights and potential therapies for these debilitating conditions.

Advisor:
Prof. Antonietti Paola F.,
Prof. Bonizzoni F.,
Dr. Corti Mattia

Academic year:
2021-2022

1. Alzheimer Disease

Neurodegenerative disorders are a complex of neurological diseases that consist of the progressive damage of the neuronal tissue: this damage first compromises communication between cells and then the entire cell structure, eventually leading to death. They are defined as *degenerative* because nerve cells do not replicate and, apart from the presence of a limited number of progenitor cells, have no possibility of being replaced. This is why one speaks of irreversible damage to neurons. Within the following study, we focus on the most well known and widespread degenerative disorder: Alzheimer's Disease (AD). AD is biologically attributable to the formation of plaques containing β-amyloid - a peptide originating from Amyloid Precursor Protein - and neurofibrillary tangles containing the τ protein - abbreviated from tubulin associated unit [6].
Generally, Alzheimer's arises as a short-term memory deficit, but it is quite common for other functions, such as language or attention, to be affected as well. The behavioural symptoms of AD are in fact directly linked to the accumulation of these two proteins, which cause damage and destruction of the synapses that mediate memory and cognition. The loss of synapses can be caused by the inability of living neurons to maintain functional axons and dendrites or by the death of neurons [4]. Unfortunately, therapeutic efforts are still struggling to find targets that substantially alter the clinical course of people with AD, within a framework of both prevention and actual treatment.

2. Mathematical Model

Recent studies have reinforced the hypothesis that the Prion Paradigm is the mechanism underlying the progression of neurodegenerative disorders, including Alzheimer's disease. Assuming that some proteins get misfolded due to a triggering mechanism, the prion's hypothesis states that they can act as infectious agents, leading to the misfolding of healthy proteins, hence their switch to infectious agents. This process causes the aggregation of pathological proteins into clusters, which spread across the brain network and transmit a chronic disease [5, 8].
From an analytical point of view, we identify the progression of misfolded proteins in the brain with the standard Fisher-Kolmogorov (FK) equation for population dynamics.

2.1. The Fisher-Kolmogorov Model

To set the analytical environment of the study, we introduce the mathematical model used to study the kinetics of protein misfolding: the simple one-concentration Fisher-Kolmogorov (FK) model. FK model has a single unknown - the dimensionless misfolded protein concentration c, which is the amount of misfolded proteins divided by the total amount of proteins - and it converts healthy to misfolded proteins at a rate α - the *conversion rate*, indeed. In particular, we consider $c \in [0, 1]$, where $c = 0$ refers to healthy proteins and $c = 1$ to misfolded proteins. Moreover, we consider $\boldsymbol{D}$ the (3D) diffusion tensor. For this kind of problem, we expect that for the smallest perturbation from the healthy state, $c > 0$, all proteins will convert from the healthy to the misfolded state, $c = 1$.

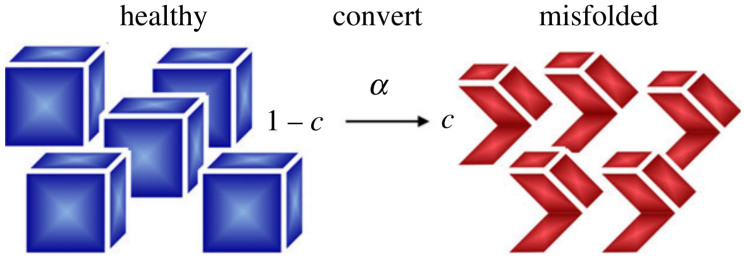

Figure 1: The Prion's Hypothesis: from healthy to misfolded proteins (source: [5])

Now we present the Fisher-Kolmogorov model. Suppose $\Omega \subset \mathbb{R}^3$, $t \in (0, T]$, $c : \Omega \times (0, T] \to [0, 1]$ and $c_0 : \Omega \to [0, 1]$. The model consists of the following three equations [3]:

$$\begin{cases} \dfrac{dc}{dt} = \nabla \cdot (\boldsymbol{D}\nabla c) + \alpha c\,(1 - c) & \forall (x, t) \in \Omega \times (0, T] \\ c(x, 0) = c_0(x) & \forall x \in \Omega \\ \dfrac{\partial c}{\partial n} = 0 & \forall x \in \partial\Omega \end{cases} \tag{1}$$

The FK model is currently the most widely used in population dynamics processes since it is simple but still accurate. As it clearly emerges from the equations above, the spreading of AD can be described by means of just two terms: the diffusion tensor $\boldsymbol{D} \in \mathbb{R}^{3\times3}$, encoding information about the movement and diffusion of the misfolded proteins within the 3D-continuous domain, and the reaction term α. Moreover, the FK equation allows a continuous representation of the brain domain in 3D and is solvable by classical finite element methods, provided suitable boundary and initial conditions are imposed. The solution that can be obtained is extremely precise at every point within the brain. In fact, by using this method, it is possible to obtain an approximate solution for the spread of the disease at every point $\boldsymbol{x}$ within the brain in time t within a given time frame, allowing for precise control over the progression of the disease. Nevertheless, the treatment of the non-linear term $\alpha c(1 - c)$ may not always be easy. Furthermore, due to the complex structure of the brain, even finite elements discretization can be very complex and computationally heavy. This is a limitation in approaching the problem from this point of view, given the computational burden of solving it.

2.2. Brain Network Models

Our goal is to develop a model that can accurately approximate a continuous system while reducing computational costs. In order to do so, we discretize the problem, dividing the domain of the system into a finite number of nodes N. Starting from the tractography of a brain and using different atlases (further details in Section 3), we generate different graphs for each patient. From there, one can create the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ (Figure 2) of every graph. This matrix represents the physical connections between the nodes in the brain, specifying among which nodes proteins can spread and how fast and massively this movement can happen. In fact, the larger the value of A_{ij}, the wider the channel between the nodes i and j. This information is crucial to understand the spread of proteins within the brain and for developing models to simulate the process.
To further reduce the complexity of the model, we compute the Laplacian Graph (see section 2.2.1), which is a matrix $\boldsymbol{L} \in \mathbb{R}^{N\times N}$ that represents the discretization of the Diffusion Tensor $\boldsymbol{D}$. The relationship between the diffusion tensor $\boldsymbol{D}$ and the Laplacian Graph $\boldsymbol{L}$ are explained in detail in (see section 2.2.2).
Once we have defined the N nodes and we have built the Laplacian Graph $\boldsymbol{L}$, we can calculate a solution function $\boldsymbol{c}(t) : [0, T] \to [0, 1]^N$. The function $\boldsymbol{c}(t)$ represents the concentration of misfolded proteins in every node of the domain and at time instant t, for $t \in (0, T]$. To complete the setting, the initial concentration of all the nodes is represented by the vector $\boldsymbol{c_0} \in \mathbb{R}^N$.
To sum up, the creation of the model involves discretizing the domain into a finite number of nodes, computing the Laplacian Graph to encode information about the movement and diffusion of the substance within the domain and calculating a solution $\boldsymbol{c}(t)$ to represent the concentration of the misfolded proteins in space ad time.

In the following subsections, we will cover the following topics:

- Creating the Laplacian Graph $\boldsymbol{L}$: we explain how to construct the Laplacian Graph from the given domain of nodes and adjacency matrix $\boldsymbol{A}$. We will describe the steps involved in this process and how the resulting graph encodes information about the Diffusion Tensor $\boldsymbol{D}$.
- The relationship between the Laplacian Graph and the Diffusion Tensor: we investigate into the details of how the Laplacian Graph and Diffusion Tensor are related, and how this relationship can be used to transition from a continuous problem to a discrete one.

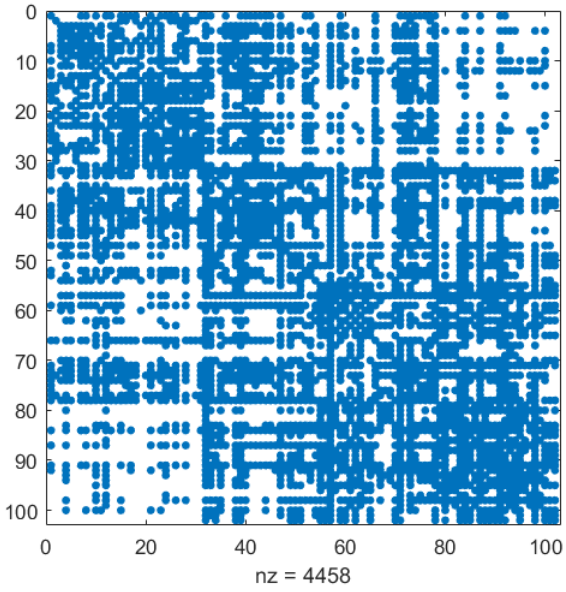

Figure 2: Example of binary adjacency graph $\boldsymbol{A}$. The blue point in the position i, j indicates if there is an edge between the node i and the node j

- The Fisher-Kolmogorov mathematical model: we introduce the FK equation on the graph, describing the key features of the model and how it works in practice.

2.2.1 Laplacian Graph

The key element at the core of the problem's discretization is a matrix called the Graph Laplacian. We are interested in deriving this matrix and its importance in the spatial discretization of the problem.

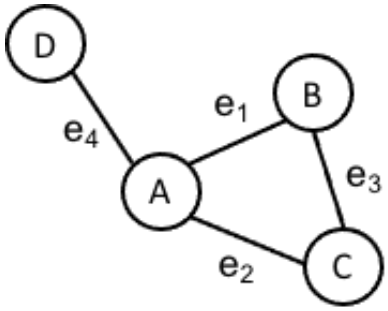

Figure 3: Example of graph (source [2])

Given a graph $\boldsymbol{G} := (\boldsymbol{V}, \boldsymbol{E})$ where $\boldsymbol{V} := \{v_1, v_2, \ldots, v_N\}$ is the set of nodes/vertices and $\boldsymbol{E} := \{e_1, e_2, \ldots, e_m\}$ is the set of edges, we define the adjacency (or connectivity) matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ (Figure 2) as

$$A_{i,j} := \begin{cases} w_{ij} & \text{if there is an edge between } v_i \text{ and } v_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $w_{ij} > 0$ is the weight of the corresponding edge, and the degree (diagonal) matrix $\mathcal{D} \in \mathbb{R}^{N \times N}$, where

$$\mathcal{D}_{i,j} := \begin{cases} \text{degree}(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

More precisely, the degree of each node represents how much a node is related to all the other nodes and it is obtained in the following way:

$$\text{degree}(v_i) = \sum_{j=1}^{N} w_{ij} \qquad \forall i \in 1, \dots, N \tag{4}$$

In our case, we only use undirected graphs, assuming that proteins can diffuse between nodes along the same channel in both directions equally. In particular, this choice on the model implies $w_{ij} = w_{ji} \quad \forall i, j \in 1, \dots, N$, leading to symmetry in all the matrices we use.
Now, we can finally define the Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{N \times N}$ as:

$$\boldsymbol{L} := \mathcal{D} - \boldsymbol{A} \tag{5}$$

In particular, using the graph in Figure 3 and supposing all the weights equal to 0 or 1, we can define the matrices as follows:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \mathcal{D} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \Longrightarrow \quad L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix}$$

At this point, it is natural to generalize these matrices to the case of a binary graph, that is a graph in which the connections are no longer associated with a weight w_{ij}. In particular:

$$A_{i,j} := \begin{cases} 1 & \text{if there is an edge between } v_i \text{ and } v_j \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Moreover, it's easy to check that each element of the diagonal of $\mathcal{D}$ is the total number of connections to the associated node. In the end, $L_{ij} = L_{ji} = -1$ if there is an edge between i and j, otherwise $L_{ij} = L_{ji} = 0$ and the diagonal of $\boldsymbol{L}$ corresponds to the diagonal of $\mathcal{D}$.

2.2.2 Relation between Laplacian Graph and Diffusion Tensor

To better understand the process of transitioning from a continuous problem to a discrete problem, it is useful to examine the analogy between the Diffusion Tensor and the Laplacian Graph. The Laplacian Graph is a matrix that encodes information about the connections between nodes in a graph, while the Diffusion Tensor is a mathematical object that describes the movement and diffusion of a substance within a domain. By analyzing the relationship between these two operators, we can rewrite the discrete mathematical model using the aforementioned Laplacian matrix $\boldsymbol{L}$.
For the purpose of this analysis, we will consider a binary (unweighted) graph, but the results can be easily generalized to the case of a weighted graph. Our goal is to understand how to use the Laplacian Graph to represent the Diffusion Tensor and solve problems on the graph in an efficient manner.
The Laplacian matrix $\boldsymbol{L}$ for a graph $\boldsymbol{G} := (\boldsymbol{V}, \boldsymbol{E})$ captures the same idea as the Laplacian for continuous, multivariate functions. We can see this by constructing all of the analogous components for graphs that are required to construct the Laplacian for continuous, multivariate functions. More precisely, we define graph operators for points, functions, gradients, and divergences. For this case, we use again as an example the previous graph in Figure 3.

Points

Each vertex $v \in \boldsymbol{V}$ in our graph will be analogous to a point in a Euclidean space.

Functions

We can define a function on the graph $f : \boldsymbol{V} \to \mathbb{R}$ mapping from each vertex to a real number, as in Figure 4:

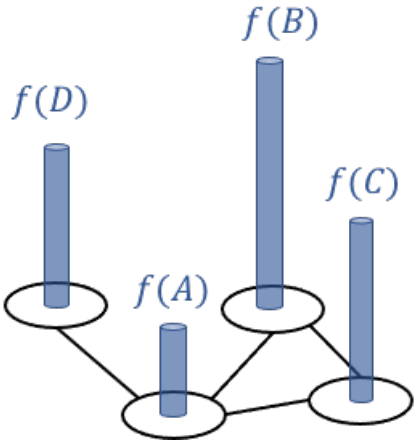

Figure 4: Function evaluated on the graph (source [2])

Ordering the vertices in the graph, $v_1, v_2, \ldots, v_N$, we can represent the function as a vector:

$$\boldsymbol{f} := [f(v_1), f(v_2), \cdots, f(v_N)] \tag{7}$$

Gradient

The analogue gradient for a graph is simply the set of edges between the vertices. The module of a gradient vector then corresponds to the difference in the values of the function between the two vertices. That is, for edge $e_k = (v_i, v_j) \in \boldsymbol{E}$ connecting node v_i to node v_j (where $i < j$ in the vertices' ordering) its "gradient" can be taken to be: $g(e_k) := f(v_i) - f(v_j)$,

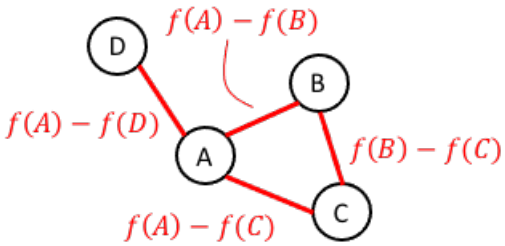

Figure 5: Gradient evaluated on the graph (source: [2])

Ordering all of the edges in the graph similarly to how we ordered all of the vertices, then we can represent the gradient as a vector:

$$\boldsymbol{g} := [g(e_1), g(e2), \ldots, g(e_m)]\,, \tag{8}$$

where m denotes the number of edges.
To conclude this first step, we need to introduce the Incidence matrix $\boldsymbol{K}$. The rows of $\boldsymbol{K}$ correspond to the sorted vertices and the columns correspond to the sorted edges. For vertex v_i and edge $e_j = (v_h, v_k)$, the entry K_{ij} is defined as:

$$K_{ij} := \begin{cases} 1 & \text{if } i = k \text{ and } i > h \\ 1 & \text{if } i = h \text{ and } i > k \\ -1 & \text{if } i = k \text{ and } i < h \\ -1 & \text{if } i = h \text{ and } i < k \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

In our example, the incidence matrix leads to the following representation (Figure 6):

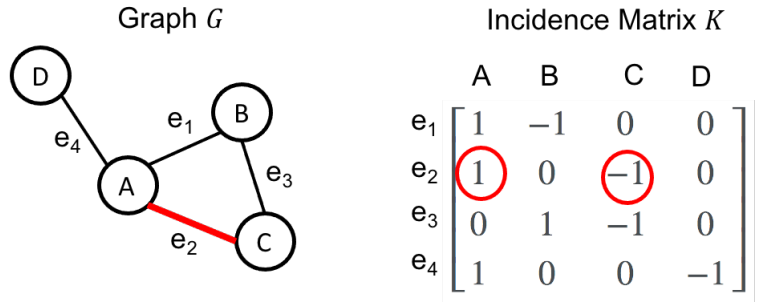

Figure 6: Incidence matrix, (source [2])

All in all, given a graph function f, we compute the "gradient" as $\boldsymbol{K}\boldsymbol{f} = \boldsymbol{g}$, as shown in Figure 7:

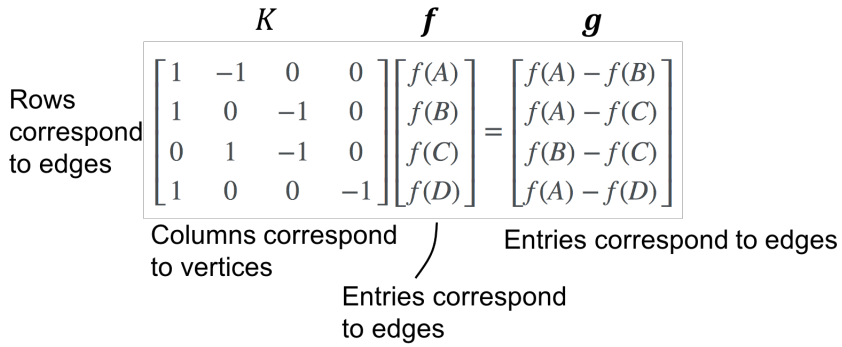

Figure 7: Gradient Calculation, (source [2])

Divergence

To understand the concept of divergence for graphs, we should consider it as the measure of "flow" in and out of each vertex, determined by the adjacent edges. This can be calculated by summing the edge values around each vertex.
For our example graph, starting from $\boldsymbol{g}$, then the divergence at node A should simply be the sum of the edge values adjacent to A where each edge value is multiplied by -1 if the flow is flowing into A. In the case of node A, all edges flow outward and thus

$$\text{div}(A) = g(e_1) + g(e_2) + g(e_4) \tag{10}$$

It's easy to check that the transpose of the incidence matrix $\boldsymbol{K}^\top$ computes the divergence, namely $\boldsymbol{K}^\top \boldsymbol{g} = \boldsymbol{d}$, as shown in Figure 8:

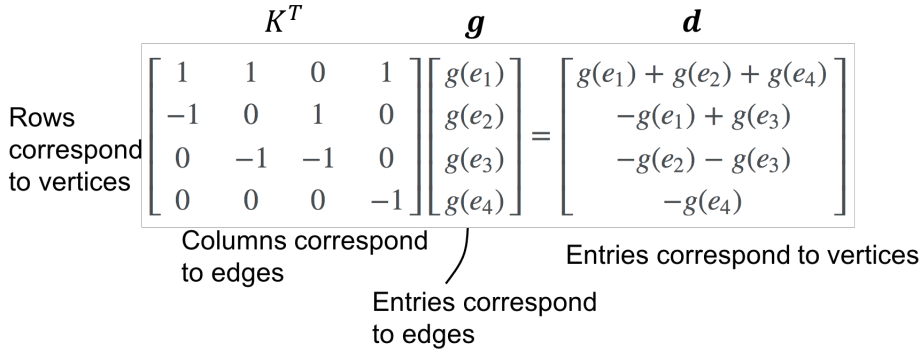

Figure 8: Divergence Calculation, (source [2])

To conclude, the Laplacian on a graph is the divergence applied to the gradient of $\boldsymbol{f}$. Applying the divergence to the gradient is carried out by the matrix product $\boldsymbol{K}^\top \boldsymbol{K}$. This is the Laplacian matrix $\boldsymbol{L}$. That is,

$$\boldsymbol{L} := \boldsymbol{K}^\top \boldsymbol{K} \tag{11}$$

If we inspect $\boldsymbol{K}^\top \boldsymbol{K}$ more closely, we can show that this matrix is exactly the degree matrix minus the adjacency matrix:

$$\boldsymbol{K}^\top \boldsymbol{K} = \mathcal{D} - \boldsymbol{A} \tag{12}$$

For more detailed explanations, see the following link: Graph Laplacian.
In particular, built the graph $\boldsymbol{G}$, we can relate the diffusion tensor $\boldsymbol{D}$ univocally with the Laplacian Graph $\boldsymbol{L}$, namely $\boldsymbol{D} \iff -\boldsymbol{L}$. In general, the Laplacian matrix is a matrix representation of a graph that encodes information about the connectivity of the graph's vertices and that is the key to jumping from a continuous problem to a more discrete model of brain networks.

2.2.3 Network equations

Discretizing the spatial domain by dividing it into a finite number of nodes and representing the connections between them using a graph, we can finally jump from the continuous Fisher-Kolmogorov model to a discrete model. Next, we write a system of N nonlinear ordinary differential equations to describe the dynamics of proteins in the nodes. In this system, the variables are the concentrations of proteins at each node at each time instant. To account for the discretization of the domain, we replace the continuous Laplacian operator with the Graph Laplacian, while leaving the reactive term and temporal derivative unchanged. Finally, we evaluate the initial condition of the system at each node. This results in a discrete model that can be used to simulate the spread of proteins within the brain and solve problems on the graph efficiently. Suppose $t \in (0,T]$, $\boldsymbol{c} : (0,T] \to [0,1]^N$ and $\boldsymbol{c_0} : \{x_1, \dots, x_N\} \to [0,1]^N$. The model consists of the following two equations [5]:

$$\begin{cases} \dfrac{dc_i(t)}{dt} = -\displaystyle\sum_{j=1}^{N} L_{ij} c_j(t) + \alpha c_i(t)\,(1 - c_i(t)) & \forall t \in (0,T] \quad \forall i \in 1, \dots, N \\ c_i(0) = c_0(i) & \forall i \in 1, \dots, N \end{cases} \tag{13}$$

We can rewrite the differential equations in matrix form:

$$\begin{cases} \dfrac{d\boldsymbol{c}(t)}{dt} = -\boldsymbol{L}\boldsymbol{c}(t) + \alpha \boldsymbol{c}(t)\,(1 - \boldsymbol{c}(t)) & \forall t \in (0,T] \\ \boldsymbol{c}(0) = \boldsymbol{c_0} \end{cases} \tag{14}$$

In conclusion, we have introduced a method for transitioning from a continuous model to a discrete model on a graph, using the Fisher-Kolmogorov model. By discretizing the spatial domain and representing the connections between nodes using a graph, we are able to write a system of differential equations that describes the dynamics of proteins within the brain.

This model will be the one we will work on subsequently and discretize in time. We will use it to run simulations on various patients and analyze the similarity with reality. It provides a useful tool for understanding the dynamics of protein diffusion and for developing models to simulate the spread of proteins within the brain.

3. Diffusion MRI Analysis and Graph Generation

3.1. Imaging through MRI

Neurodegenerative disorders like Alzheimer's disease typically spread across the brain through patterns that privilege certain areas, while other regions are affected only at a later stage. As an example, medial temporal lobe structures, entorhinal and perirhinal cortex and the hippocampus are those areas where the earliest stages of the disease are found. Thus, using imaging techniques in order to investigate these regions could help the early diagnosis of such diseases, by exploiting this typical progression pattern.
The imaging technique used for our project is MRI (Magnetic Resonance Imaging), widely available and useful to generate the graphs that will constitute our computational domain.
In particular, we take into account the MRI of three different patients:

- A healthy 42-year-old patient
- An 80-year-old patient affected by Alzheimer's disease
- A healthy 96-year-old patient

This differentiation is useful in order to analyze three different conditions and also to evaluate how their situations can impact the associated graph.

3.2. Fiber Tracking

The generation of the graph that represents the brain network is performed using *DSI studio* [1], a tractography software tool that, starting from DTI-MRIs, permits to map brain connections.
The first step consists in generating the whole-brain tractography by fiber tracking. Starting from the MRI and using*DSI Studio*, it is possible to obtain the tractography presented in Figure 9.

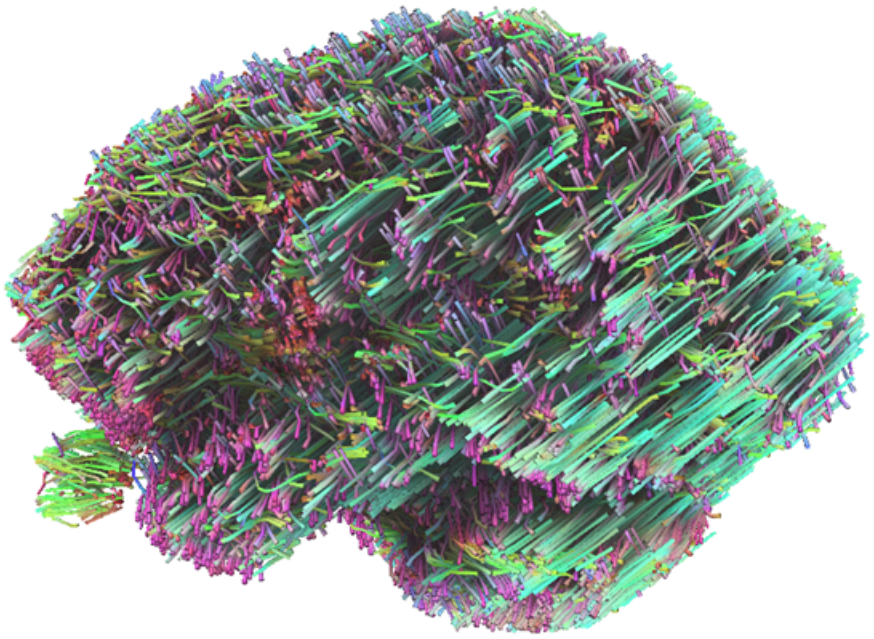

Figure 9: Example of a whole-brain tractography generated using fiber tracking.

This kind of image describes the mapping of the location and direction of white matter bundles and their constituent fibers within the human brain. The key passage is to take into account the necessary precision in terms of reconstructed fibers in order to obtain a good network. This step is crucial to have a quantitative description of the amplitude of connections between different brain areas.

3.3. Graph Generation

At this point, the generation of the graph can be performed. In order to do so, it is necessary to evaluate and choose the right atlas that permits obtaining a good reduced model in terms of the number of nodes of the graph. The aim, indeed, is finding a good trade-off between accuracy and complexity. Each atlas, indeed, divides brain regions in different ways, leading to graphs with different structures and numbers of nodes. To perform a good evaluation we decided to analyze the properties of the network associated with the atlas. Apart from the number of nodes, other important indicators are, for instance, diameter, density and clustering.
This analysis leads us to identify four particularly interesting atlases: *Brainnectome*, *CerebrA*, *FreeSurferSeg* and *HCP-MMP*. Their main characteristics are represented in Table 1.

Properties	Brainnectome	CerebrA	FreeSurferSeg	HCP-MMP
Number of nodes	246	102	192	360
Diameter	4	3	3	4
Density	0.346872	0.495826	0.242419	0.210198
Clustering	0.697402	0.768486	0.701483	0.676552

Table 1: Main atlases and corresponding properties

At this point, the graph associated with each of the four chosen graphs is generated, as shown in Figure 10.

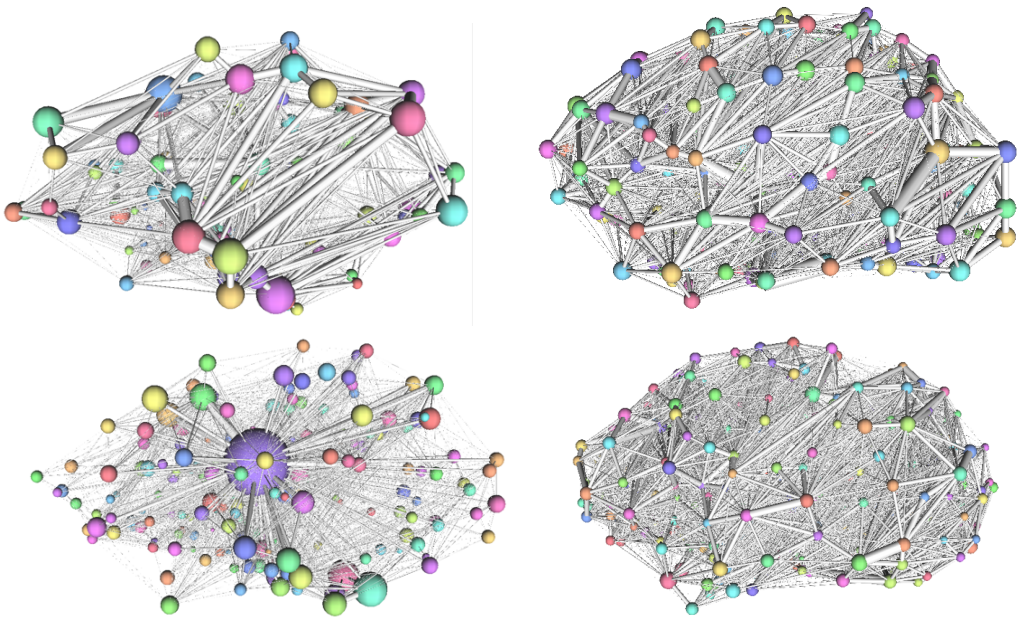

Figure 10: Graphs associated to the following atlases: *CerebrA* (top-left), *Brainnectome* (top-right), *FreeSurferSeg* (bottom-left) and *HCP-MMP* (bottom-right).

Each graph is characterized by its connectivity matrix A, a $N \times N$ matrix (where N is the number of nodes), where each element A_{ij} contains the normalized weight (see Section 2.2.1, (2)) of the arc that connects node i to node j. This is the matrix that we used inside our $MATLAB$ code.

3.4. Setting the Initial Concentration

Finally, in order to proceed with the numerical implementation of the problem, we first need to understand from which nodes of the graph the diffusion starts. From a medical point of view, this means understanding which are the nodes that correspond to areas such as the entorhinal cortex, where the neurodegenerative process usually begins.

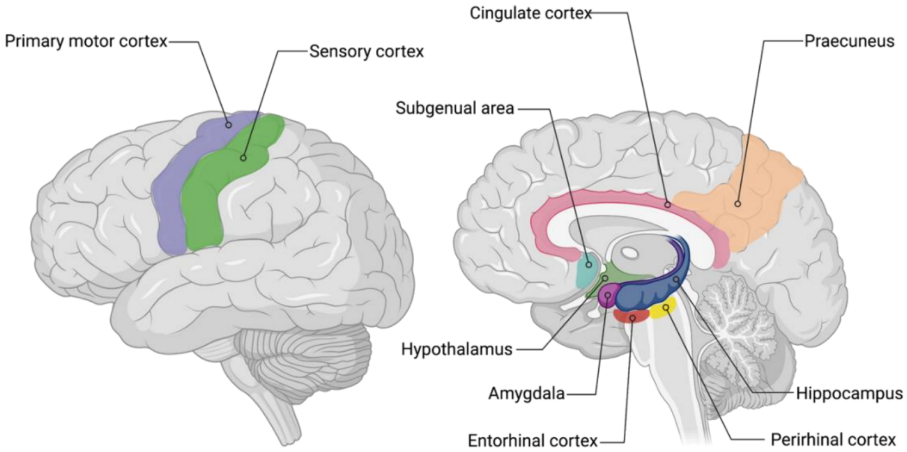

Figure 11: Different regions of the brain and corresponding functions. In particular, the entorhinal cortex is where the triggering mechanism of AD begins (source [7])

In order to do so, the computation of the barycenters (presented in Appendix A) of each of the brain regions identified by the atlases is crucial. For the *CerebrA* atlas the identification is easy, since the nodes associated with the entorhinal cortex are explicitly tagged, and they are the 4^{th} and the 55^{th} node.
The idea, at this point, is to exploit the fact that same regions, on different atlases, should have almost the same coordinates. Obviously, this is an approximation, but we can still assert that it can reflect quite well the truth.
Starting from this, we aim to identify which were the nodes on the other atlases with the same coordinates as these two nodes. To do so, we write the `Python` code presented in Appendix B.
In conclusion, once the coordinates of all the two entorhinal nodes are extracted (for each atlas), we decide to set to 0.5 the concentration there, while for all the other nodes the concentration is set to zero.

4. Numerical Discretization

The numerical discretization we have applied in this study departs from classical finite elements and instead looks toward graph-based discretization. The profoundly complex nature of the domain - i.e. the brain - is of fundamental importance in the choice of the discretization method: the brain tissue in fact represents a highly irregular domain, as well as a two-dimensional surface located within a three-dimensional space. Moreover, the medical problem itself is not limited to the surface of the cerebral cortex, but analyses elements inside the brain: the neurons and the network that connects them. Finally, the choice of graphs to approximate the brain is mainly determined by their extremely lower computational cost with respect to finite elements. It is sufficient to think that the anatomical-cerebral atlases we use do not exceed 360 nodes (Figure 12(c)), whereas a finite-element treatment would require millions degrees of freedom (Figure 12(b)), at least.

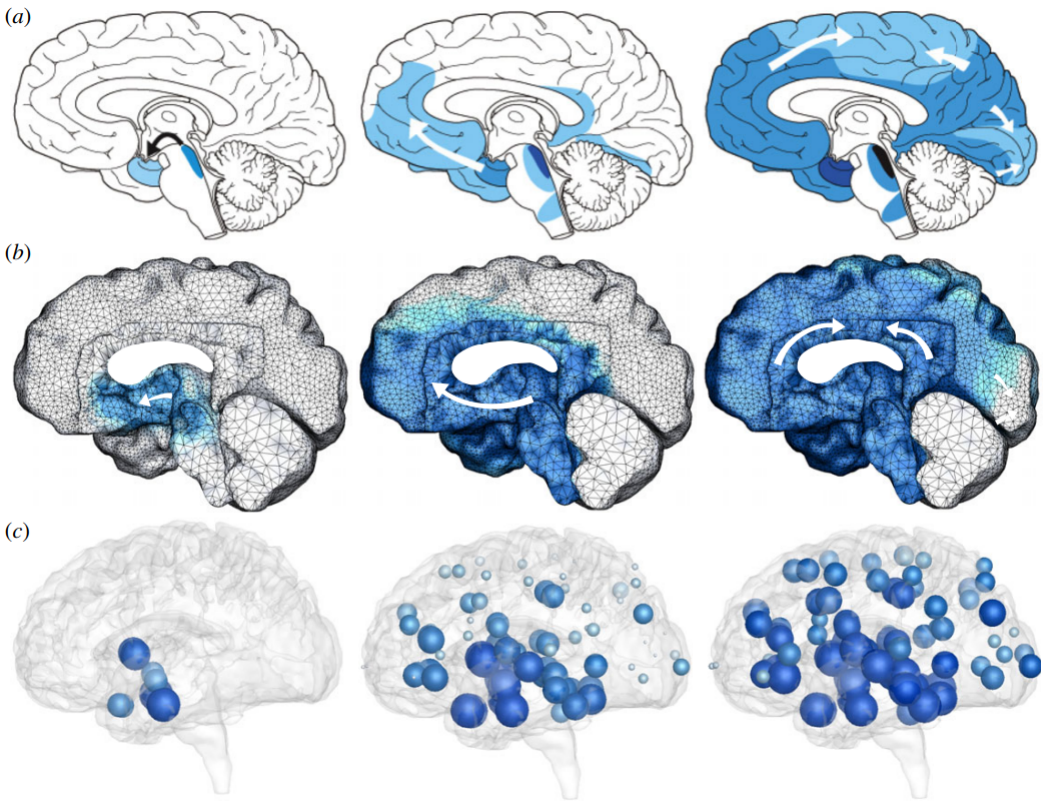

Figure 12: Comparison between the number of degrees of freedom in approximating the brain network by means of finite elements and of a graph. Clearly, there is a considerable computational advantage in applying a graph approximation.

However, the switch of perspective from standard finite elements to numerical analysis on graphs is not trivial, hence it needs to be studied in depth. In the next paragraphs we deal with very simple geometries in order to focus only on the numerical methods and on the differences and similarities between the two approaches. In particular, we will validate the methods and we will propose some numerical approaches to obtain optimality in convergence orders.

4.1. Validation of the Space Discretization Scheme

4.1.1 From the Graph to 1D Finite Elements

To properly verify and validate our PDE model, it is crucial to have a strong understanding of PDEs on graphs and their associated analysis of convergence, stability and consistency. Unfortunately, our current knowledge in this area is not as well-established as it is for PDEs on 1D and 2D meshes. Without a solid theoretical foundation, it would be difficult to accurately measure the effectiveness of numerical methods for PDEs on graphs. To address this issue, we shift our focus to a 1D finite element approach, where there is a wealth of precise results in the literature. However, it should be noted that the relationship between a graph and a 1D grid is not straightforward. To simplify the process, we use a toy problem in which the graph is represented by four circularly connected nodes. We then duplicate the top-left node, "cut" the graph and stretch it along the positive real semi-axis, resulting in a 1D mesh in the interval [0,1] with four equally spaced intervals (Figure 13).
Before analyzing the numerical methods for both the graph and 1D mesh, the exact solutions used for validation must meet a specific requirement. To ensure consistency between the analysis on the graph and on the 1D mesh, the boundary conditions, which for simplicity we consider to be Dirichlet's, must simulate the periodicity of the graph in space. This means that $c(0,t) = c(1,t) \quad \forall t$. If this condition was not met, there would be a discontinuity in the solution at the top-left node of the graph, which is not a plausible scenario. The strategy used in Figure 13 may appear to be intuitive, but it is necessary

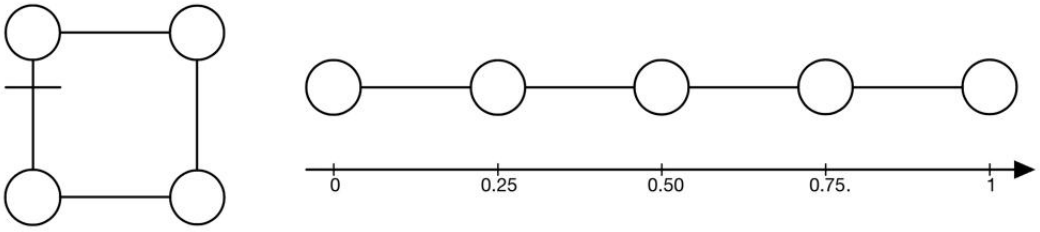

Figure 13: From a circular graph to 1D finite elements

to analyze the numerical structure of both cases to investigate the possibility of relating them in a rigorous way.
As we are dealing with a diffusion equation, the analysis focuses on the Laplacian matrices of the graph and of the 1D grid. Let $\boldsymbol{L}$ be the Laplacian matrix related to the graph (Subsection 2.1) and $\boldsymbol{S}$ be the tridiagonal matrix related to second order derivatives. For this specific problem, they take the following form:

$$\boldsymbol{L} = \begin{bmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix} \qquad \boldsymbol{S} = -\frac{1}{h^2} \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

where h denotes the mesh size of the one-dimensional discretization.
It is then evident that they share a quite similar pattern, where only two main differences arise:

- the matrix $\boldsymbol{S}$ presents the multiplicative term $-\frac{1}{h^2}$, relating precisely to the 1D mesh. This difference is natural to emerge since we work on a one-dimensional grid by successive refinements. Notice that the elements of $\boldsymbol{L}$ and $\boldsymbol{S}$ have opposite sign because the Laplacian matrix $\boldsymbol{L}$ encodes the operator $-\Delta$ on the graph.
- elements $\boldsymbol{L}_{14}$ and $\boldsymbol{L}_{41}$ differ from $\boldsymbol{S}_{14}$ and $\boldsymbol{S}_{41}$, respectively: this difference identifies precisely the fact that the graph is connected, while the mesh, by construction, is not. In this regard, homogeneous Dirichlet's boundary conditions turn out to be ideal to recover this difference. This approach will allow us to switch from the matrix $\boldsymbol{L}$ to $\boldsymbol{S}$, hence to finite elements.

4.1.2 The Finite Elements Toy Problem

Once we have clarified the transition from graph-based treatment to the 1D finite element approach, we face the following toy problem, which is based on the equation (14):

$$\begin{cases} \dfrac{\partial c}{\partial t} = \dfrac{\partial^2 c}{\partial x^2} + \alpha c\,(1-c) + f & \forall x \in (0,1), \forall t \in (0, 2\pi], \\ c(x,0) = c_0(x) & \forall x \in [0,1], \\ c(0,t) = c(1,t) = 0 & \forall t \in [0, 2\pi], \end{cases} \tag{15}$$

As far as discretization is concerned, we consider h the spatial step and N the total amount of spatial intervals (i.e. $N = \frac{1}{h}$), Δt the temporal step and K the total amount of time intervals (i.e. $K = \frac{2\pi}{\Delta t}$), $c_i^k \simeq c(ih, k\Delta t)$ the approximation of the exact solution and $f_i^k = f(ih, k\Delta t)$ the evaluations of the forcing term. Furthermore, we adopt finite forward differences for time discretization and the θ-method. If we denote $\boldsymbol{S}_1$ as the i^{th} row of the second derivative matrix, what we obtain is:

$$\begin{cases} \dfrac{c_i^{k+1} - c_i^k}{\Delta t} = (1-\theta)\boldsymbol{S}_i c_i^k + \theta(1-\theta)\boldsymbol{S}_i c_i^{k+1} + \\ \qquad +\alpha\left[(1-\theta)c_i^{k+1} + \theta c_i^k\right](1-c_i^k) + (1-\theta)f_i^k + \theta f_i^{k+1} & \forall i = 1, ..., N-1, \forall k = 1, ..., K, \\ c_i^{k=0} = c_0(x = ih) & \forall i = 0, ..., N, \\ c_{i=0}^k = c_{i=N}^k = 0 & \forall k = 0, ..., K, \end{cases} \tag{16}$$

There are a few preliminary remarks to present in order to better understand the validation stage.

- To approach the convergence study of a numerical method, it is necessary to test it on a known exact solution. For this reason, the external forcing term f is introduced: this allows us to choose a function c to approximate and fit the term f in order to make it an exact solution of the diffusion equation being studied.
- The boundary conditions imposed are homogeneous Dirichlet's: this is a standard procedure and it also meets the aformentioned requirement, that is reproducing periodicity in space of the solution on the graph.
- Currently, the nonlinear term is treated in a semi-implicit manner. However, a more in-depth discussion on the nonlinear term of the equation will be discussed in Section 4.2.2.

4.1.3 Order of Convergence with respect to Space

Once clarified these necessary premises, let us consider as exact solution the function

$$c(x,t) = \sin(t)\sin(2\pi x) \qquad \forall x \in [0,1], \quad \forall t \in [0,2\pi] \tag{17}$$

that verifies homogeneous Dirichlet conditions and satisfies problem (15) with forcing term

$$f(x,t) = \frac{dc}{dt} - \frac{d^2c}{dx^2} - \alpha c(1-c) \qquad \forall x \in [0,1], \quad \forall t \in [0,2\pi]. \tag{18}$$

Within this section, we investigate the effect of the spatial discretization step h on the convergence order. The time discretization step is kept constant as the spatial refinements progress for two reasons. Firstly, this allows us to obtain results that are independent of both the space and time variables. Secondly, simulations on real graphs consider the Laplacian matrix as a datum of the problem, so this matrix is constant and encodes all the information of the problem, such as the graph's geometry, connection weights, and three-dimensional nature of the domain. Therefore, the actual problem reduces to approximating an ODE, which is studied in Section 4.2.
To evaluate the spatial convergence rate, we conduct 6 consecutive refinements of the 1D mesh, beginning with $h = 0.25$ ($N = 4$ spatial intervals) and ending with $h = 0.0078125$ ($N = 4 \times 2^6 = 128$). The number of time intervals, $K = 10000$, is chosen such that it does not affect convergence, ensuring independence from the time variable.
Finally, some relevant plots follow.

- First of all, in Figure 14 we plot the numerical solution at each node in the domain at the last refinement ($N = 128$) as a function of time. It is quite evident that it approximates precisely the exact solution $c(x,t) = \sin(t)\sin(2\pi x)$. The color legend indicates the coordinate of the node being plotted.
- We then focus on the 4 original nodes of the graph, which are the locations where we are interested to approximate the solution. In Figure 15 we plot the corresponding numerical solutions refinements advances. We can see that, although the first refinement overestimates the numerical solution, the convergence to the exact one is evident.
- Finally, in Figure 16 we inspect the order of convergence of the solution and it turns out that, as we expect, we get $\|c - c_h\|_\infty \sim O(h^2)$, where c_h denotes the numerical approximation of c.

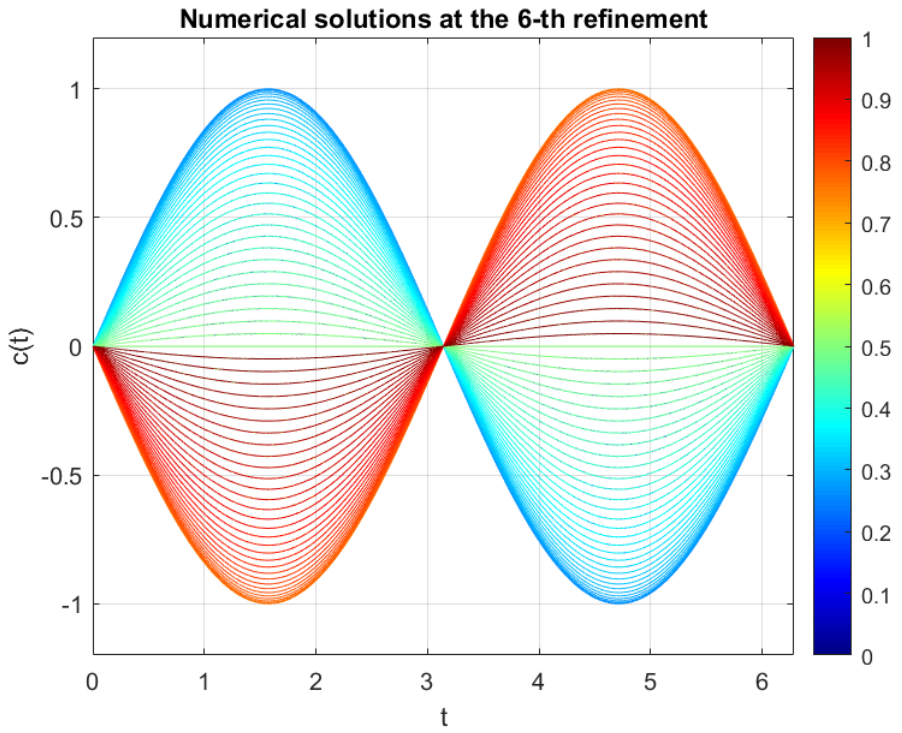

Figure 14: Numerical solution in each discretization point at the last consecutive refinement

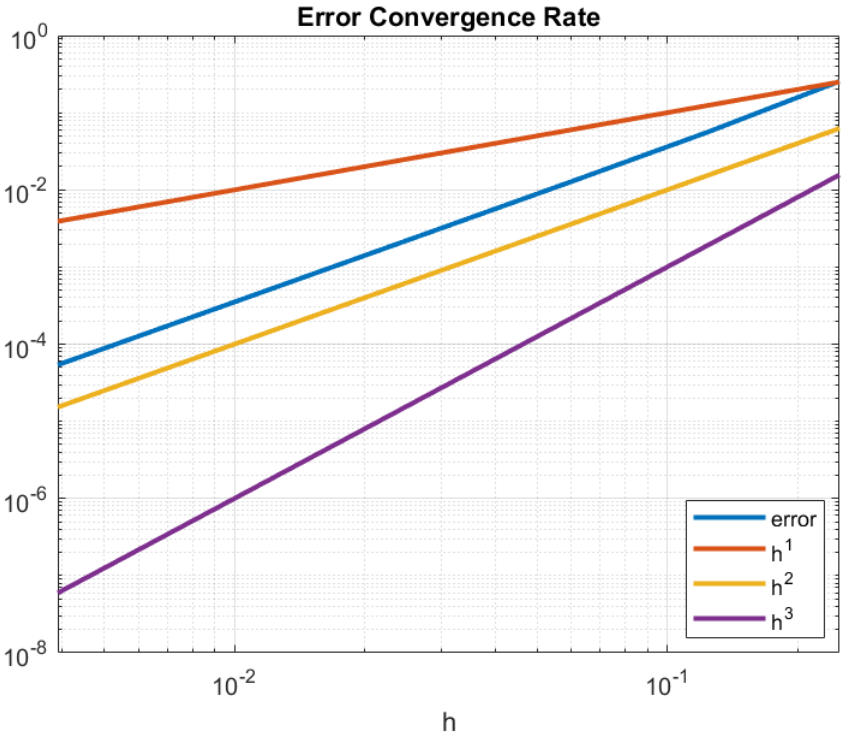

Figure 16: Convergence orders with respect to the mesh size h

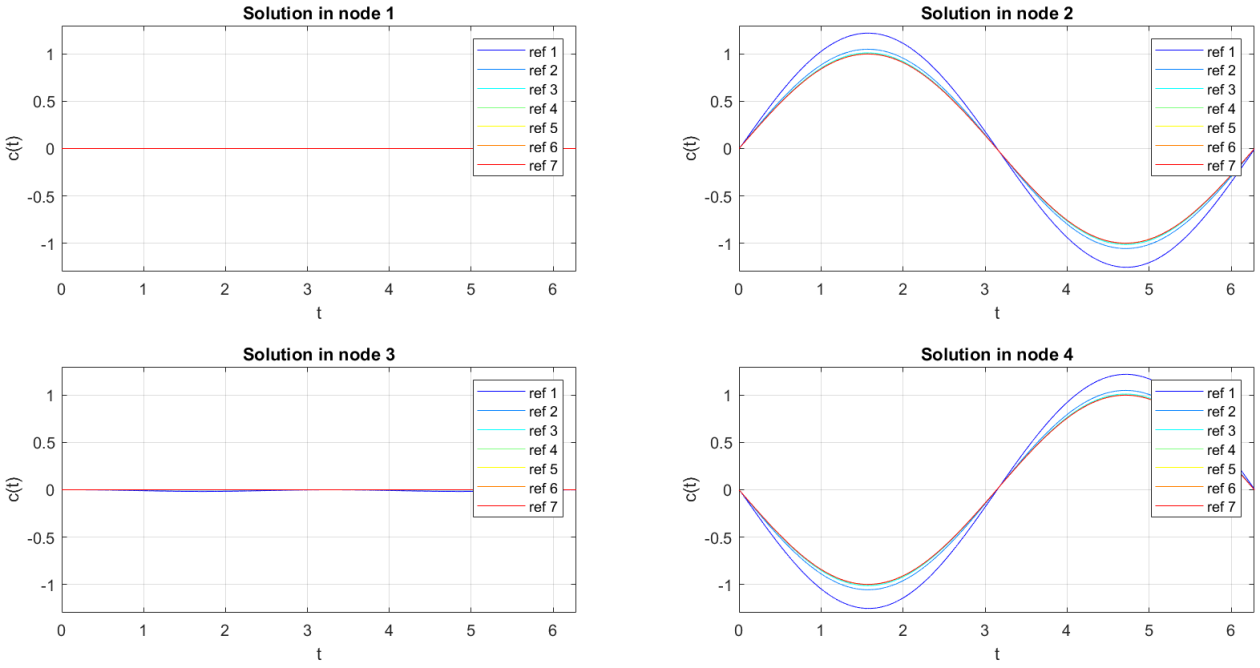

Figure 15: Numerical solution in the 4 original nodes. For each node, every refinement is plotted in order to observe convergence

4.2. Validation of the Time Discretization Scheme

4.2.1 The Toy Problem

Once the spatial discretization has been studied, we move on to the validation of the method for what concerns the temporal discretization. In fact, as mentioned earlier, at this stage of the study we are no longer interested in the Laplacian matrix $\boldsymbol{L}$, which is for all intents and purposes a datum of the problem. Instead, now we compare our results to the network model 2.2.3, rather than the continuous one used in the previous phase of code validation (Section 4.1). The problem, therefore, becomes a system of nonlinear ODEs, whose unknown is

$$\mathbf{c} = \mathbf{c}(t) = \begin{bmatrix} c_1(t) \\ c_2(t) \\ \dots \\ c_N(t), \end{bmatrix}, \tag{19}$$

where N is the number of nodes in the graph and $\mathbf{c}$ depends uniquely on time. These equations are not independent, but are related by the Laplacian matrix $\boldsymbol{L}$, that contains all information about the domain of the problem, i.e. the graph.
To proceed with the analysis of the method, let us set up the following toy problem: consider again a graph of 4 nodes, connected in a circular fashion. Thus, in this case, we obtain a system of 4 nonlinear ODEs, each one with the corresponding initial condition:

$$\begin{cases} \dfrac{d\mathbf{c}}{dt} = -\boldsymbol{L}\mathbf{c} + \alpha\mathbf{c}\,(1-\mathbf{c}) + \mathbf{f} & \forall i = 1,2,3,4, \quad \forall t \in (0, 8\pi], \\ \mathbf{c}(0) = \mathbf{c}_{0,i} & \forall i = 1,2,3,4, \end{cases} \tag{20}$$

where $\mathbf{c}_{0,i}$ the initial condition of the i^{th}-equation. Again the term $\mathbf{f} = \mathbf{f}(t)$ appears, which, in total analogy with the discretization of the PDE, is present to ensure that the function $\mathbf{c}(t)$ we choose is a solution of the system in question. Finally, note that the equations cannot be solved independently since they are in mutual relation via the matrix $\boldsymbol{L}$.

Before moving forward with the formulation of the discrete problem, it is useful to anticipate a result that will be analyzed below. At this stage, the challenge is the numerical approximation of the nonlinear term $\alpha c(1-c)$: in order to overcome it, the θ-method is applied to the first term, and the second term is dealt with implicitly (details of this can be found in Section 4.2.2). By now, to simplify notation, we use $\Psi[\mathbf{c}(1-\mathbf{c})]$ to represent the discretization of the nonlinear term. Additionally, we define Δt as the time step, K as the total number of time intervals (i.e. $K = \frac{8\pi}{\Delta t}$), $c_i^k \simeq c_i(k\Delta t)$ and $f_i^k = f_i(k\Delta t)$. Lastly, $\boldsymbol{L}_i$ denotes the i^{th} row of the Laplacian matrix $\boldsymbol{L}$. At this point it is possible to proceed with the formulation of the discrete problem:

$$\begin{cases} \dfrac{c_i^{k+1}-c_i^k}{\Delta t} = -(1-\theta)\boldsymbol{L}_i c_i^k - \theta \boldsymbol{L}_i c_i^{k+1} + \alpha\Psi[\mathbf{c}(1-\mathbf{c})] + \\ \qquad\qquad +(1-\theta)f_i^k + \theta f_i^{k+1} \qquad\qquad \forall i = 1,2,3,4 \quad \forall k = 1,...,K \\ c_i^0 = c_i(t=0) \qquad\qquad \forall i = 1,2,3,4, \end{cases} \tag{21}$$

4.2.2 Treatment of the Non-Linear term

The Fisher equation, with its nonlinear term $\alpha c(1-c)$, characterizes the evolution of a population, in our case, misfolded proteins.
We observe that the first factor c falls entirely within the θ-method, which, however, does not affect the second factor $1-c$. The latter, in fact, is treated in an implicit manner. This approach is called *Semi-Implicit Treatment* of the nonlinear term with I order extrapolation ($c_i^{k+1} \simeq c_i^k$)

$$\Psi[\mathbf{c}(1-\mathbf{c})] = \left[(1-\theta)c_i^k + \theta c_i^{k+1}\right](1-c_i^k) \qquad \forall i = 1,2,3,4 \quad \forall k = 1,...,K. \tag{22}$$

The θ-method is commonly used for linear equations and, for those, it is well-known that it achieves order of convergence 1 for $\theta = 0, 1$ and order of convergence 2 for $\theta = 0.5$. However, this result is not immediate for nonlinear equations and, although the former approach is satisfactory, the non-linearity of the problem does not allow us to gain one order of convergence in the case $\theta = 0.5$.
Therefore, we decide to take action on the discrete formulation using a II order extrapolation ($c_i^{k+1} \simeq \frac{3}{2}c_i^k - \frac{1}{2}c_i^{k-1}$):

$$\Psi[\mathbf{c}(1-\mathbf{c})] = \left[(1-\theta)c_i^k + \theta c_i^{k+1}\right]\left[1 - \left(\frac{3}{2}c_i^k - \frac{1}{2}c_i^{k-1}\right)\right] \tag{23}$$

Evidently, the initial conditions of the discrete problem become:

$$\begin{cases} c_i^0 = c_i(0) \qquad \forall i = 1,2,3,4, \\ c_i^1 = c_i(\Delta t) \quad \forall i = 1,2,3,4, \end{cases} \tag{24}$$

Finally, this approach proves successful in increasing the order of convergence for $\theta = 0.5$, a behavior we analyze in detail in the next section (Figure 17).

4.2.3 Order of Convergence with respect to Time

In this section we undertake a quantitative study of convergence orders with respect to the time discretization step Δt. We first choose an exact solution at each of the 4 nodes. Given the nonlinear term $\mathbf{c}(1-\mathbf{c})$, the method is stable only at exact solutions $\mathbf{c}(t) \in [-1, 1]$. Furthermore, since $\mathbf{c}$ is a quantity indicating the concentration of diseased proteins within the brain network, it is necessarily subject to the physical constraint $\mathbf{c} \in [0, 1]$. Given these considerations and defining $T = 8\pi$ we choose:

$$\begin{cases} c_1(t) = \frac{1}{2}\sin(t) + \frac{1}{2} & \forall t \in [0,T] \\ c_2(t) = \frac{1}{2}\cos(t) + \frac{1}{2} & \forall t \in [0,T] \\ c_3(t) = \frac{1}{T}t & \forall t \in [0,T] \\ c_4(t) = \frac{1}{T}t\left(\frac{1}{2}\cos(t) + \frac{1}{2}\right) & \forall t \in [0,T] \end{cases} \tag{25}$$

As a consequence, we obtain $\forall t \in [0, T]$:

$$\begin{cases} f_1(t) = \frac{3}{2}\sin(t) - \frac{t}{T} + \frac{1}{2}\left[\alpha\left(\sin(t)+1\right)\left(\frac{1}{2}\sin(t) - \frac{1}{2}\right)\right] - \frac{1}{2T}\left[t\left(\cos(t)+1\right)\right] + 1 \\ f_2(t) = \frac{3}{2}\cos(t) - \sin(t) - \frac{t}{T} + \frac{1}{2}\left[\alpha\left(\cos(t)+1\right)\left(\frac{1}{2}\cos(t) - \frac{1}{2}\right)\right] - \frac{1}{2T}\left[t\left(\cos(t)+1\right)\right] + 1 \\ f_3(t) = 3\frac{t}{T} - \frac{1}{2}\sin(t) - \frac{1}{2}\cos(t) + \frac{1}{T} - \frac{1}{2T}\left[t\left(\cos(t)+1\right)\right] + \frac{1}{T}\left[\alpha t\left(\frac{t}{T} - 1\right)\right] - 1 \\ f_4(t) = \frac{1}{2T}\left(\cos(t)+1\right) - \frac{1}{2}\sin(t) - \frac{t}{T} - \frac{1}{2}\cos(t) - \frac{1}{2T}t\sin(t) + \frac{3}{2T}t\left(\cos(t)+1\right) + \\ \quad + \frac{1}{2T}\left[t\left(\frac{1}{2T}\left[\alpha t\left(\cos(t)+1\right)\right] - 1\right)\left(\cos(t)+1\right)\right] - 1 \end{cases} \tag{26}$$

For the study of the convergence error, we impose 5 successive refinements of the temporal mesh, starting from $K = 6400$ time intervals and getting to $K = 6400 \times 2^5 = 206080$. In the Figure 17 we observe how the *Semi-Implicit* treatment of the nonlinear term leads to an error's rate of convergence $\|\mathbf{c} - \mathbf{c}_h\|_\infty \sim O(\Delta t)$, while by means of the second approach we obtain $\|\mathbf{c} - \mathbf{c}_h\|_\infty \sim O(\Delta t^2)$.

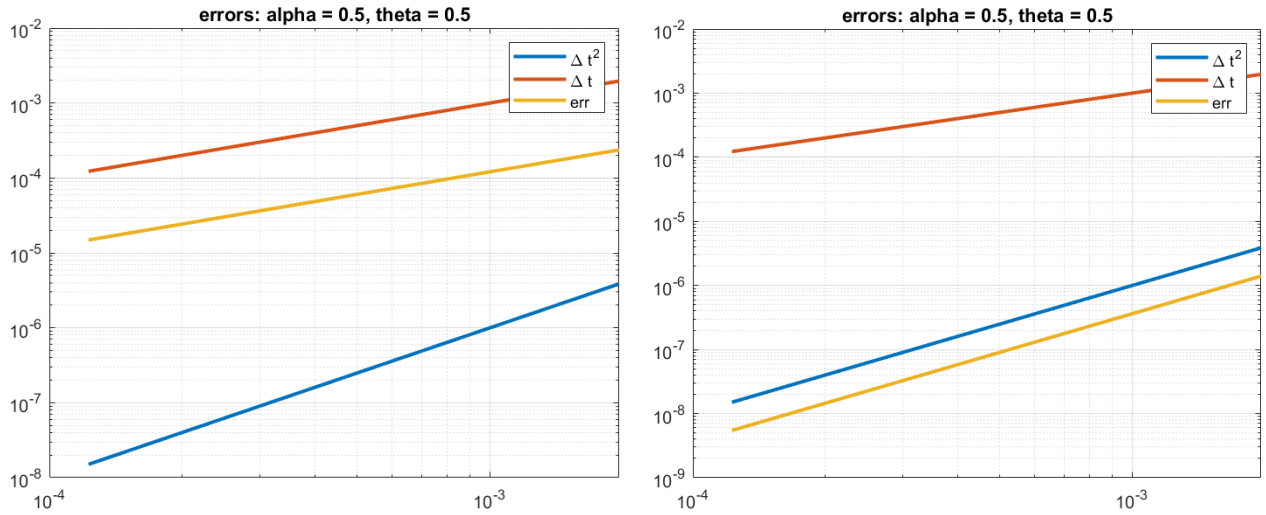

Figure 17: Convergence Orders of Semi-Implicit I order extrapolation (left) and Semi-Implicit II order extrapolation (right) treatment with respect to the time step Δt

Finally, in Figure 18 we plot the exact solution and the approximate solution for each reference node at the last time refinement, namely $\Delta t = \frac{8\pi}{206080} = 0.00012$.

4.3. Discretization on patient-specific graphs

As extensively analyzed in the previous paragraphs, we were able to prove the consistency of the code and ensure that the methods we intend to use have expected orders of convergence. After validating the numerical methods for spatial and temporal discretization, we can now focus on discretizing the diffusion problem in a much more complicated domain. In this paragraph, and in the next section, we will analyze the discretization on patient-specific graphs and draw conclusions about the simulations we will perform. As a result of our validation study, we propose the following numerical approximation method. Let the continuous-time network model (Section 2.2.3) be:

$$\begin{cases} \dfrac{d\boldsymbol{c}(t)}{dt} = -\boldsymbol{L}\boldsymbol{c}(t) + \alpha\boldsymbol{c}(t)\left(1 - \boldsymbol{c}(t)\right) \quad \forall t \in (0, T] \\ \boldsymbol{c}(0) = \boldsymbol{c_0} \end{cases} \tag{27}$$

where N is the number of nodes of the graph, $\boldsymbol{c} : [0, T] \to [0, 1]^N$ and $\boldsymbol{c_0} : \{x_1, \ldots, x_N\} \to [0, 1]^N$. Then, the most efficient discretization method we obtain in light of the validation study is the following: $\forall i = 1, 2, 3, 4 \quad \forall k = 1, ..., K$

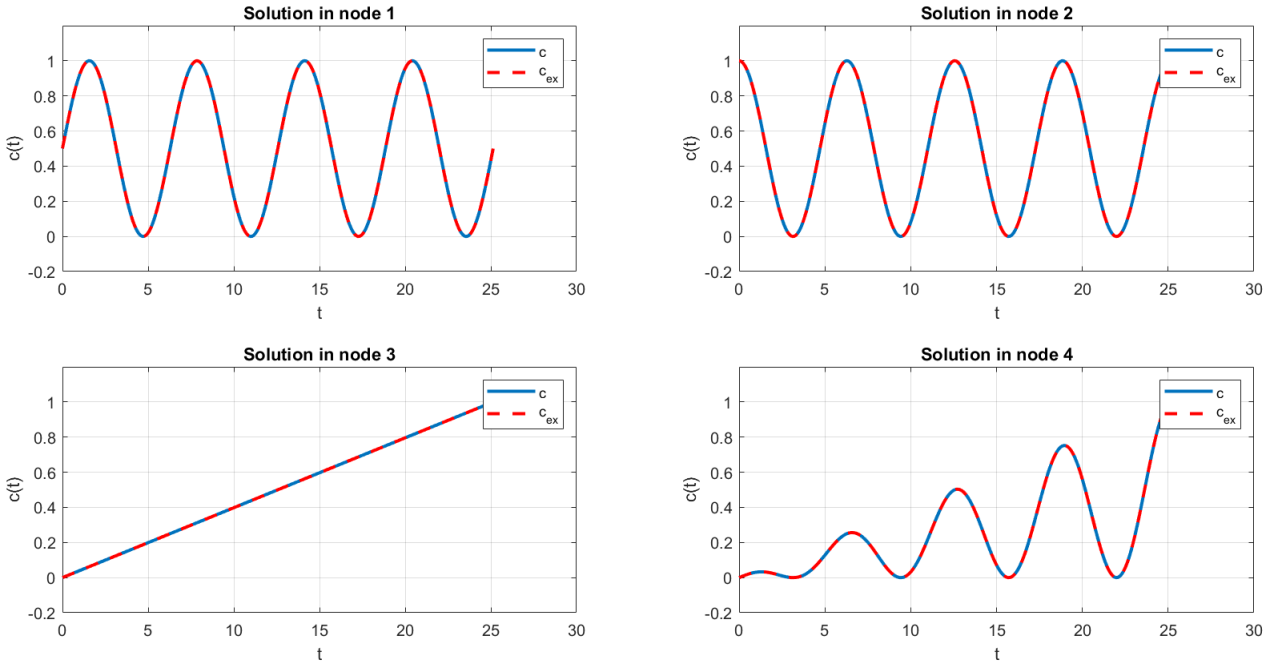

Figure 18: Time evolution of the exact and numerical solution in the 4 original nodes. The two plots overlap, meaning that the approximation is reliable.

$$\begin{cases} \dfrac{c_i^{k+1} - c_i^k}{\Delta t} = -(1-\theta)\boldsymbol{L}_i c_i^k - \theta \boldsymbol{L}_i c_i^{k+1} + \alpha \left[(1-\theta)c_i^k + \theta c_i^{k+1}\right] \left[1 - \left(\dfrac{3}{2}c_i^k - \dfrac{1}{2}c_i^{k-1}\right)\right] \\ c_i^0 = c_i(0) \\ c_i^1 = c_i(\Delta t) \end{cases} \tag{28}$$

where, in analogy with the notation employed throughout the whole project, Δt is the time step, K the total number of time intervals, $c_i^k \simeq c_i(k\Delta t)$ and $f_i^k = f_i(k\Delta t)$.

5. Simulations and Results

For the last part of our project, once the numerical discretization is performed, we run some simulations in order to study the spreading of misfolded proteins across the brain network and understand the difference in the diffusion process between the three patients that we take into account, varying the atlas among the ones presented in Section 3.3.

5.1. A preliminary simulation

First of all, we perform a simulation in which we can verify that the diffusion process follows the behavior depicted in Figure 19, obtained by Fornari [5].
In order to do that, we take into account the 42-year-old healthy patient and we choose the *CerebrA* atlas. Then, we run the simulation fixing the parameters $\alpha = 0.5$ and $\theta = 0.5$, while as final time we set $T = 50$ (in years). A subdivision of the different brain areas is also performed: this allows us to understand how the diffusion process evolves over time, by looking at the trend of misfolded proteins concentration in each area. For the following simulations, then, the brain is subdivided into 4 different regions (up-right, up-left, down-right and down-left), as shown in Figure 20.

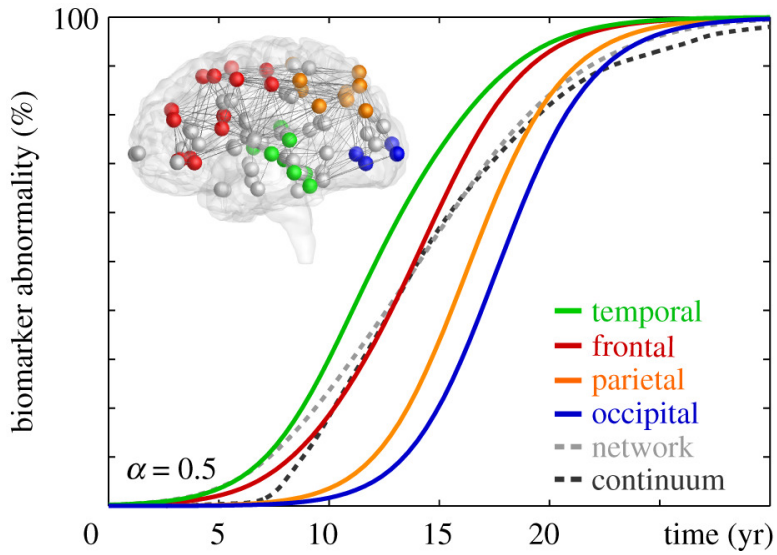

Figure 19: Evolution of the concentration of misfolded proteins obtained by Fornari [5] over 30 years.

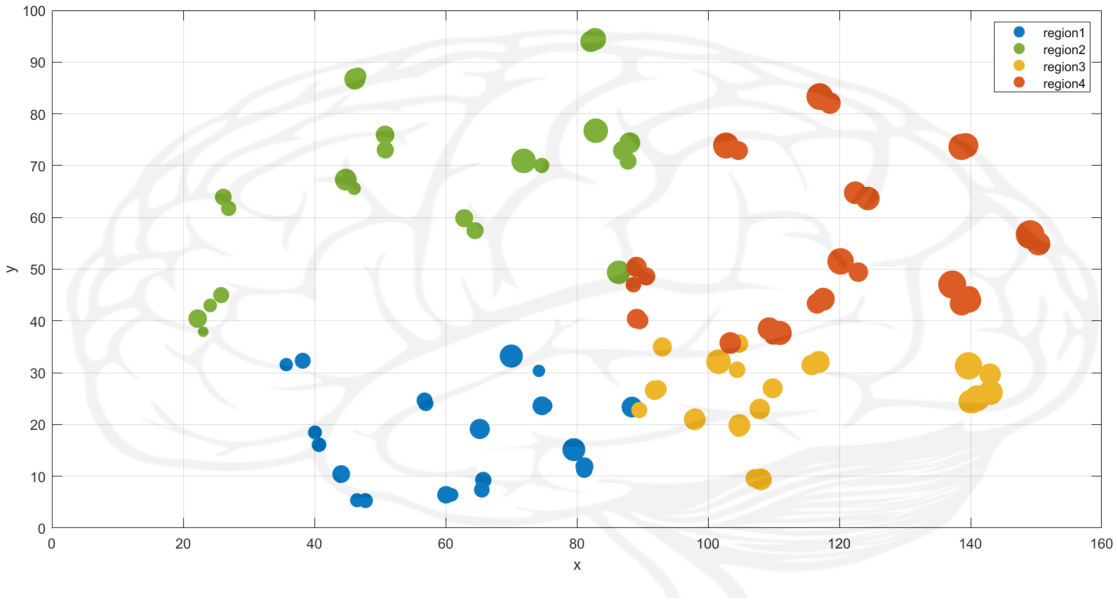

Figure 20: Distribution of nodes across the brain for the graph generated starting from the *CerebrA* atlas.

As explained in Section 3.4, the initial concentration is set as different from 0 only on two nodes, which are the ones associated with the entorhinal cortex. These nodes, in our case, are inside the yellow region (Region 3). Thus, in the first years of the simulation we expect to have a concentration different from zero only in this area.
The result of the simulation is presented in Figure 21.
As expected, we can see that the diffusion starts from Region 3 and then proceeds clockwise to Region 1 and to the other regions. This is coherent with the results presented by Fornari (Figure 19), which subdivided the brain in a similar way and the concentrations vary in accordance with the related brain region. Moreover, the time at which the saturation is reached (only misfolded proteins are present) is similar, and it is around 30 years.
At this point, another simulation is performed. We are in the same framework as the previous case, but this time we let α vary between 0 and 1. Since α is the conversion rate, this is helpful to verify that the larger the value of α the faster the growth of misfolded proteins' concentration. For larger

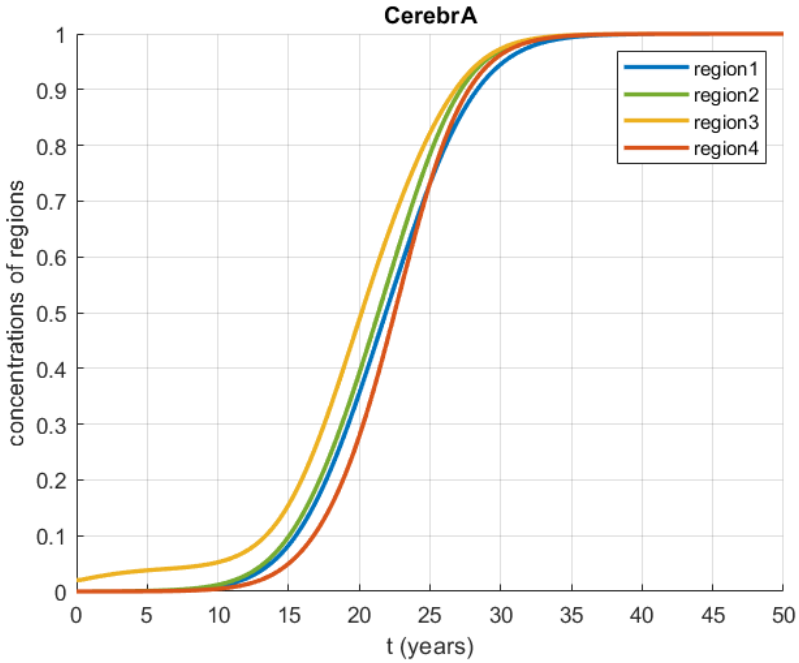

Figure 21: Evolution of the concentration of misfolded proteins in each of the four regions for a time interval of 50 years

α, indeed, the disease diffusion process is more rapid and saturation occurs earlier. The results of this simulation are presented in Figure 22, and they perfectly reflect what we expect: the higher α, the earlier saturation happens.

5.2. Different atlases comparison

In this section, we focus on the analysis of the different atlases. The framework is similar to the one presented in the previous paragraph: we consider the 42-year-old patient, a final time of $T = 50$ (years), a diffusion coefficient $\alpha = 0.5$ and $\theta = 0.5$ for the numerical method. We then perform the same simulation on the four graphs associated with the four atlases presented in Section 3.3. The results are the following.
As we can see from Figure 23, to each atlas corresponds a graph that describes brain connections in different ways. This implies that the diffusion process will be simulated differently, based on the atlas chosen, as shown in Figure 24. The simulation of the diffusion process is still valid for each of the four atlases (since they reflect the results of Fornari's paper), but the time to reach the saturation and the speed of the diffusion depends on the specific graph. In particular, *FreeSurferSeg* leads to a diffusion that is way faster than the one related to $HCP - MMP$: while in the first case the saturation is reached after around 30 years, in the latter the concentrations saturate only after 40 years.
Actually, since the differences between the results are present but are still modest, they may also be related to the division of the brain that we performed. We divided, indeed, the nodes in 4 regions using two perpendicular hyperplanes, thus leading to a subdivision that already gives a good description of the phenomenon but that it is still very simple and not specific.
Therefore, we can say that our numerical simulation, apart from some slight differences, performs well also when we change atlas (i.e. the network that represents our computational domain).

5.3. Analysis of the three patients

We now want to simulate the diffusion process for the three patients in order to observe similarities and differences. Each simulation is performed by setting a final time of $T = 50$ years, $\alpha = 0.5$ and

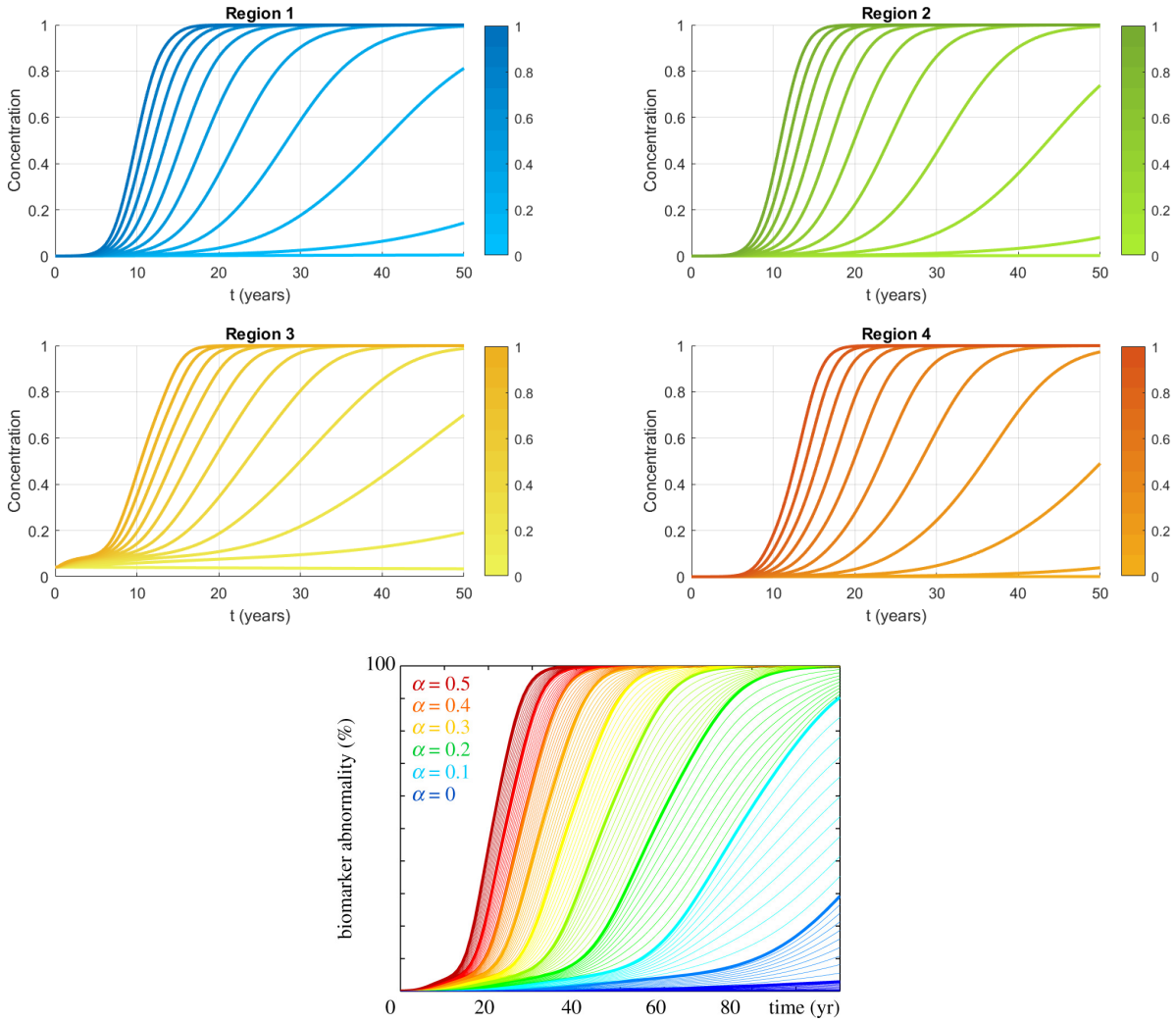

Figure 22: Evolution of the concentration of misfolded proteins varying α from 0 to 1. On top, the result obtained with our simulations, on the bottom the Fornari's one [5].

$\theta = 0.5$, and by using the *CerebrA* atlas. Obviously, such time interval is not appropriate for patients with an age above 80, but it is still meaningful to test the robustness of the numerical methods while changing the connectivity matrix, i.e. by considering different patients. The results are presented in Figure 25.

The three simulations lead to an evolution of the concentration that is similar in each case, thus highlighting the fact that the code is robust as we wanted. We should now notice that in each simulation the order of activation of brain regions is still in accordance with the results presented by Fornari, recalled in Section 5.1. However, some differences can be noticed. In the case of the 42-year-old patient the concentration of Region 1, the one in which the diffusion process starts, tends to remain always higher than the concentrations in the other regions during the simulation. For the older patients, instead, the concentration of Region 1 is higher only in the first phases of the process, and after a certain time the other regions start to present more misfolded proteins. This is exactly what we expect: for older patients, the spreading of misfolded proteins should be, indeed, accentuated. The different behavior of the diffusion process in each patient is related to their brain connectivity. Each individual, indeed, has connections among brain regions that depend on factors like age and the healthiness of the subject, thus leading to different brain networks. These peculiarities can be visualized in Figure 26, where we can notice that each patient has a network with its own characteristics, thus leading to different simulations.

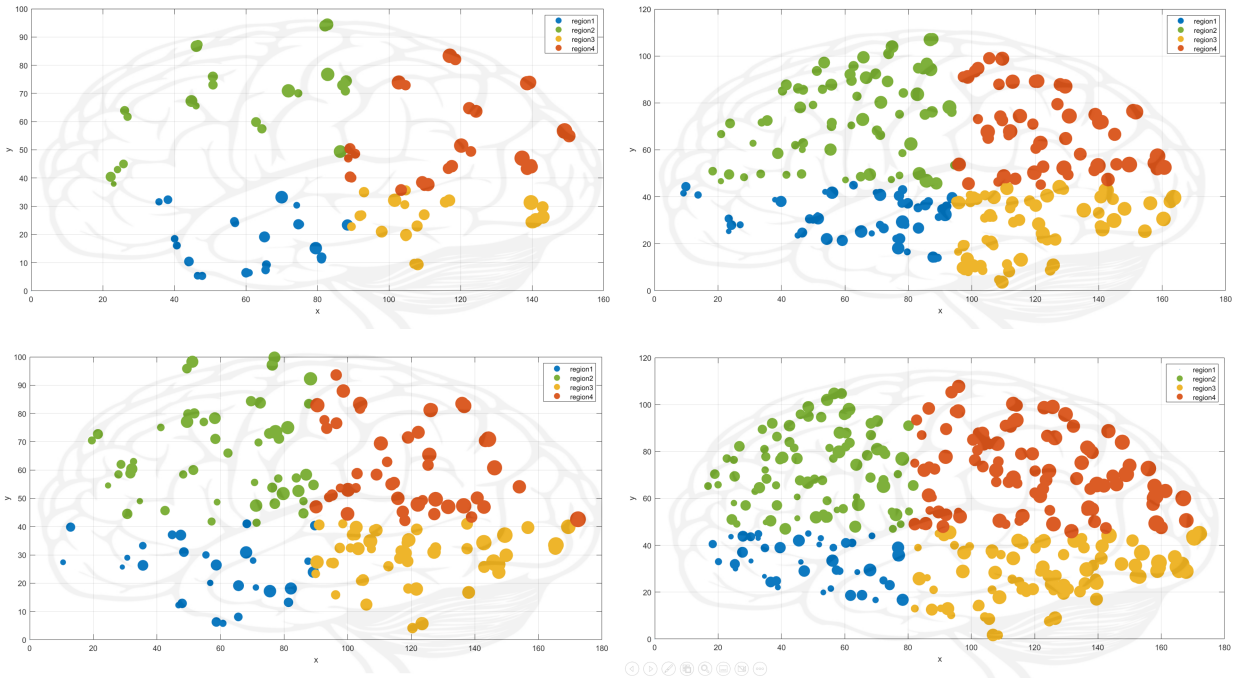

Figure 23: Distribution of nodes across the brain and their subdivision in regions for each graph associated to the relative atlas. In particular: *CerebrA* (top-left), *Brainnectome* (top-right), *FreeSurferSeg* (bottom-left) and *HCP-MMP* (bottom-right).

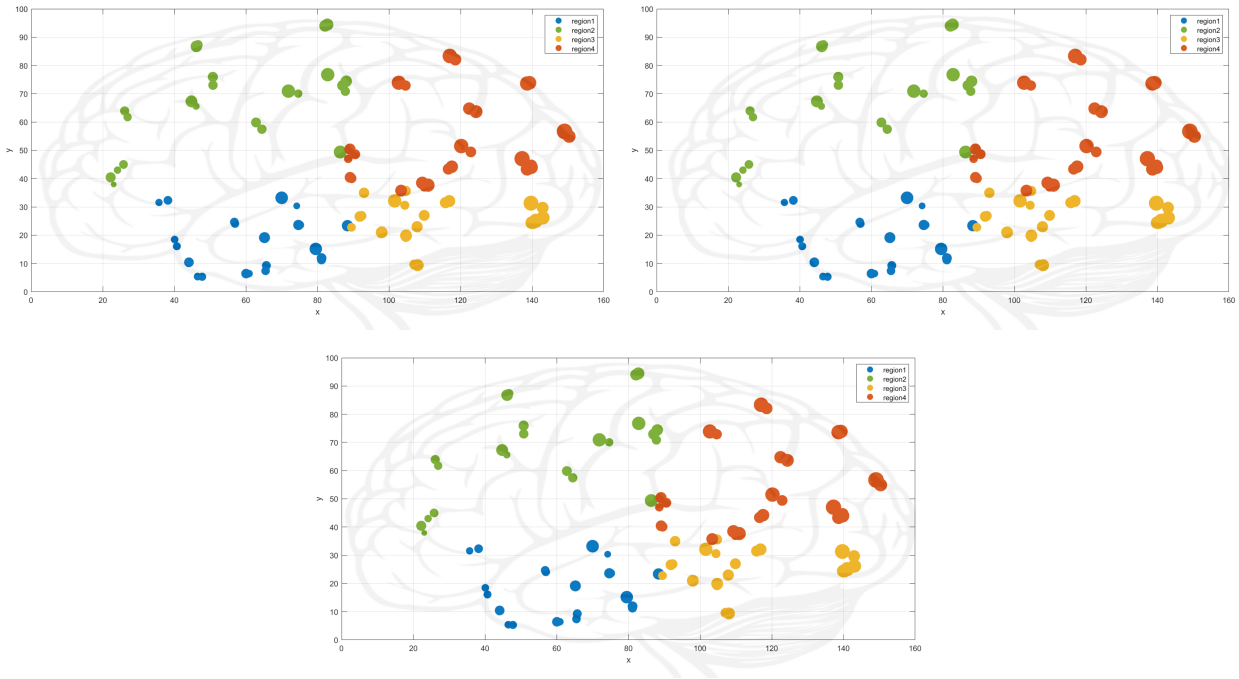

Figure 26: Distribution of nodes across the brain and their subdivision in regions for each of the three patients. In particular: 42-year-old healthy patient (top-left), 80-year-old ill patient (top-right), 96-year-old healthy patient (bottom).

5.4. MATLAB notebook

To facilitate the running of the simulations, we decided to create a MATLAB notebook in which the user can select the patient, choose the preferred parameters and run the simulations easily. The idea

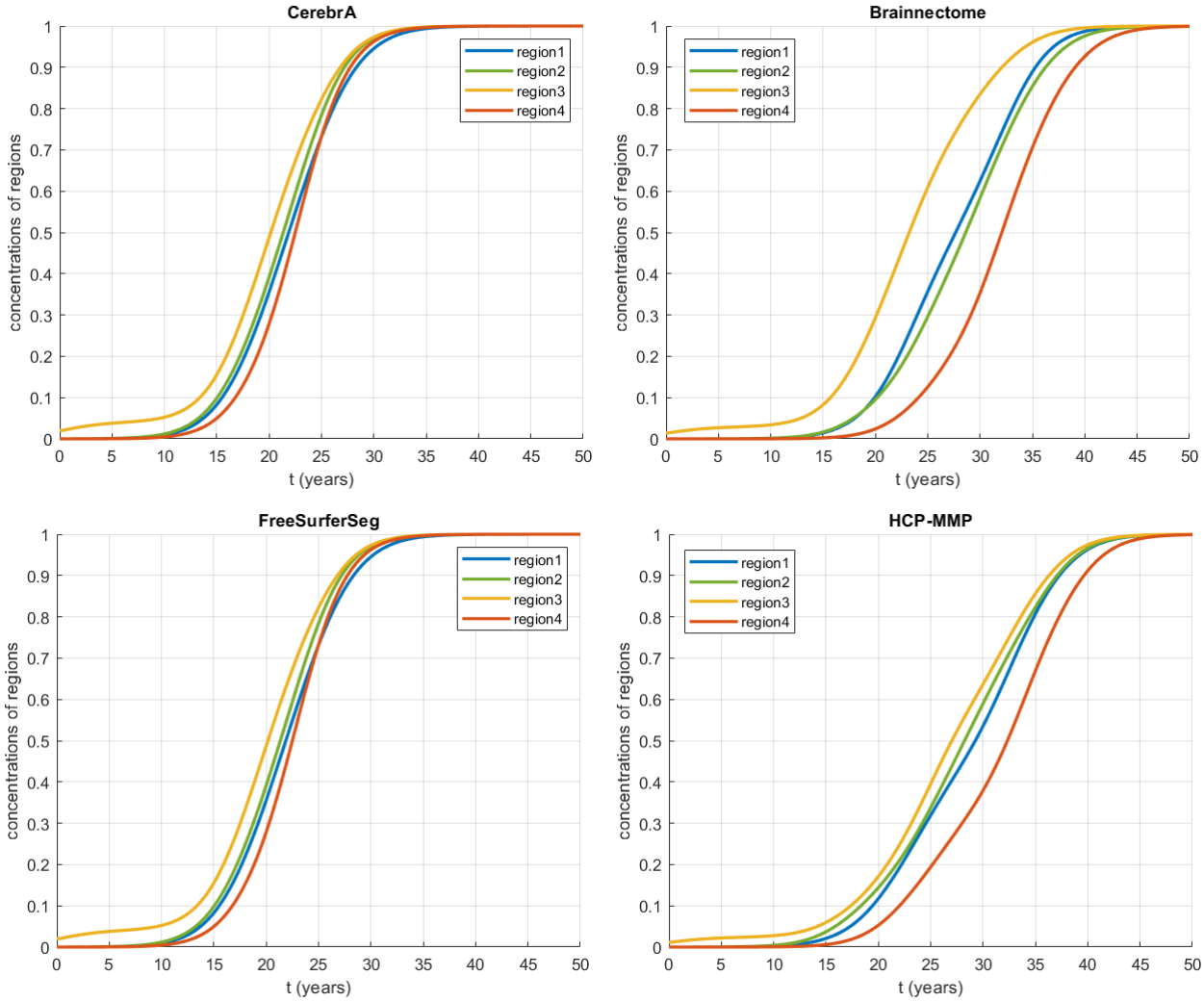

Figure 24: Evolution of the concentration of misfolded proteins varying the atlas

is to let the user be able to perform the desired simulation, without the necessity of working with the source code. This is also a way to maintain all the results and plots in the same environment, with the possibility to recover them at any time. All the simulations carried out in Section 5 are presented in the notebook.

6. Conclusions and future work

Despite it being complicated to make general observations on such a wide topic as modeling a neurodegenerative pathology, the considerations presented in this study are plausible and satisfactory. In an attempt to draw some conclusions, the following reflections have been thought throughout the whole project and are considered meaningful for a more general view of the research field in which we have engaged in depth.

- Firstly, we found the mathematical modeling approach using graphs to be both innovative and effective, proving to be an excellent compromise between modeling precision and reduced computational cost. For this reason, it would be extremely interesting to expand the study to more complex models of competition between populations. In particular, the paper from which we were most inspired, Fornari's [5], also addresses the problem using the Heterodimer and Smoluchowski models. The first (Figure 27 left) considers two different possible configurations of the protein, healthy and misfolded, introducing not only the conversion parameter, but also the clearance parameter: this approach, therefore, leads to a greater flexibility of the model and a

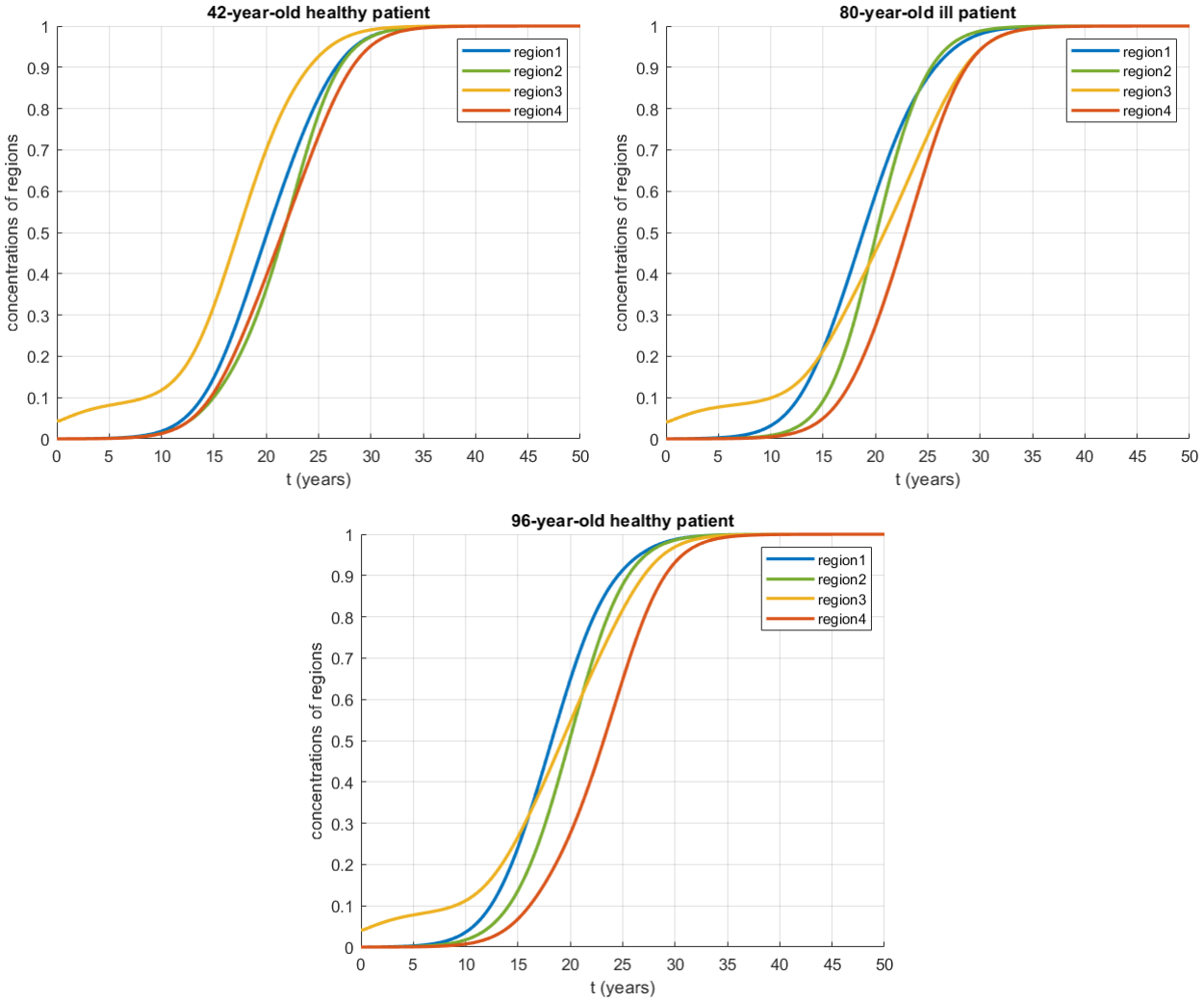

Figure 25: Evolution of the concentration of misfolded proteins for different patients.

resulting wider adherence to reality. As the number of variables increases, so does the complexity of the model. The second one, the Smoluchowski model (Figure 27 right), adds further information to the dynamics of the diffusion process: it takes into account the presence of protein clusters, which diffuse differently based on their size. The increase in complexity of this model is significant as, in parallel with the size-specific treatment of the diffusion of the misfolded proteins, several new parameters are added (such as specific transport, aggregation, fragmentation and clearance) which increase the risk of overfitting.

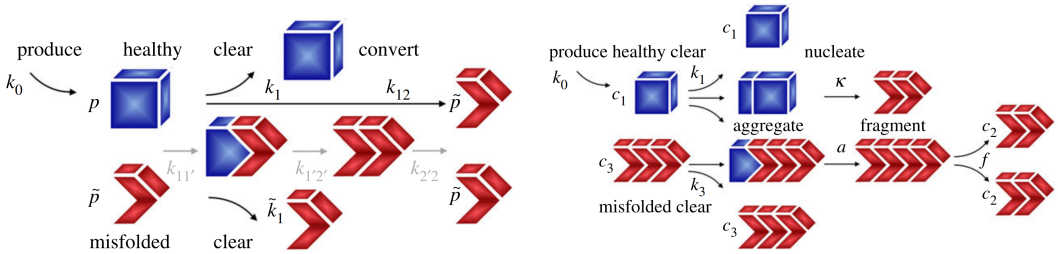

Figure 27: Heterodimer model (left) and Smoluchowski model (right) (source: [5])

- Secondly, during the project we have often thought that it would be interesting to introduce a non-negligible factor of neurodegenerative diseases. The data that are available are snapshots of the specific situation of each patient when the MRI is performed. However, we suppose that,

especially in the case of ill patients, the brain tissue, and therefore its connections, degenerate due to aging and, above all, the disease. As a consequence, it would be meaningful to model the changes in the brain graphs of patients over time. Mathematically, this coincides with the treatment of a time-varying domain, a problem that is definitely non-trivial. In this regard, however, we thought it might be interesting to propose a solution. Given the impossibility of collecting data distributed over time (the progress of the disease happens, fortunately, over many years), it would make sense to try to remove the connections between the nodes of the brain proportionally to the concentration of misfolded proteins. This would allow us to encode the degeneration of the domain in the connectivity matrix of the graph. Despite the fact that this is probably a naive proposal, the idea of introducing a degenerative factor into the model intrigues us.

In conclusion, we would like to emphasize how stimulating it has been to work on such a delicate and current topic as biomedical research. The numerical approach to modeling neurodegenerative disorders has allowed us to experience the versatility of mathematical tools, as well as their power. This adds up to the extraordinary charm of biomedical subjects, which facilitates engagement and spontaneous interest in carrying out research. In light of this, we would like to thank Professor Quarteroni and Dr. Regazzoni for the opportunity to carry out this project. Finally, the most heartfelt thanks go to Professor Antonietti, Professor Bonizzoni and Dr. Corti, our supervisors, whose availability, helpfulness and professionalism accompanied us throughout the project: to them we owe the knowledge acquired in the field of graphs and mathematical modeling of neurodegenerative diseases, as well as a deeper insight in Fisher-Kolmogorov equation.

A. Barycenters Computation

The computation of the barycenters of each of the brain regions identified by the atlases is useful for two reasons:

- A better visualization of the graph
- To set the initial concentration, as explained in the next paragraph

To do so, we first visualize the whole-brain region extracted from *DSI Studio* in *ParaView*. Then, by using the *ParaView* filter `IntegrateVariables`, we are able to compute the coordinates of the barycenter of each of the brain regions of the atlas. In particular, the extraction of the coordinates is performed using a `Python Script` that, using a for-loop, permits us to cycle over all the regions and to obtain the respective barycenter coordinate.

B. Python code to identify crucial nodes

For each atlas the following code computes the distance of every node from the two *CerebrA* nodes, and returns the coordinates of the two nodes as the *argmin* of the distances' vector.

```
import numpy as np
import scipy.io

mat_old = scipy.io.loadmat('baric_cerebra.mat')
mat_new = scipy.io.loadmat('baric_brainnectome.mat')
mat_new2 = scipy.io.loadmat('baric_freesurf.mat')
mat_new3 = scipy.io.loadmat('baric_hcp.mat')

old = np.array(mat_old['baric_cerebra'])
q = np.array([old[3], old[54]])

p = np.array(mat_new['baric_brainnectome'])
p2 = np.array(mat_new2['baric_freesurf'])
p3 = np.array(mat_new3['baric_hcp'])

def with_indices_2(p, q):
    rows, cols = np.indices((p.shape[0], q.shape[0]))
    distances = np.sqrt(np.sum((p[rows, :] - q[cols, :])**2, axis=2))
    return distances

# distances Nx2
distances = with_indices_2(p, q)
distances2 = with_indices_2(p2, q)
distances3 = with_indices_2(p3, q)

# extract neighbours
neighbours = np.argmin(distances, axis = 0)
neighbours2 = np.argmin(distances2, axis = 0)
neighbours3 = np.argmin(distances3, axis = 0)
```

References

[1] DSI Studio. https://dsi-studio.labsolver.org/.

[2] The graph laplacian. https://mbernste.github.io/posts/laplacian_matrix/, 2020.

[3] Multiscale modeling of dementia: From proteins to brain dynamics, 2022.

[4] George S Bloom. Amyloid-β and tau: the trigger and bullet in Alzheimer disease pathogenesis. *JAMA neurology*, 71(4):505–508, 2014.

[5] Sveva Fornari, Amelie Schäfer, Mathias Jucker, Alain Goriely, and Ellen Kuhl. Prion-like spreading of Alzheimer's disease within the brain's connectome. *Journal of the Royal Society Interface*, 16(159):20190356, 2019.

[6] David S Knopman, Helene Amieva, Ronald C Petersen, Gäel Chételat, David M Holtzman, Bradley T Hyman, Ralph A Nixon, and David T Jones. Alzheimer disease. *Nature reviews Disease primers*, 7(1):1–21, 2021.

[7] Wieke M. van Oostveen and Elizabeth C. M. de Lange. Imaging techniques in Alzheimer's disease: A review of applications in early diagnosis and longitudinal monitoring. *International Journal of Molecular Sciences*, 22(4), 2021.

[8] Johannes Weickenmeier, Mathias Jucker, Alain Goriely, and Ellen Kuhl. A physics-based model explains the prion-like features of neurodegeneration in Alzheimer's disease, Parkinson's disease, and Amyotrophic Lateral Sclerosis. *Journal of the Mechanics and Physics of Solids*, 124:264–281, 2019.