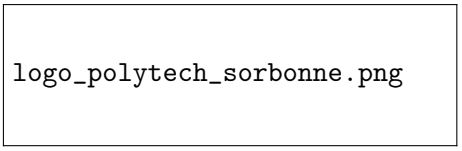

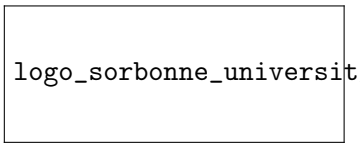

Sorbonne Université

Rapport de stage

main 5

Numerical simulations on the Smoluchowski equations

Paola ALLEGRINI

Encadrant:
Giacomo DIMARCO

July 24, 2019

Contents

Part I

Microscopic model for the aggregation and diffusion of β-amyloid peptide

1 Introduction

The Smoluchowski equation is a system of partial differential equations which describes the evolving densities of diffusing particles that are prone to coagulate in pairs. Here, we consider a set of Smoluchowski's discrete diffusion–coagulation equations to describe the aggregation and diffusion of β-amyloid peptide ($A\beta$), a process associated with the development of Alzheimer's disease.
It is well known that $A\beta$ peptide plays an important role in the process leading to neuronal death. The presence of $A\beta$ in the cerebral tissue is normal but by unknown reasons we can observe in some neurons an imbalance between production and clearance of $A\beta$ during aging. At elevated levels, it produces pathological aggregates : an accumulation of deposits known as senile plaques.
The mathematical model considered comes from ref paper. The model presented describes the aggregation and diffusion of β-amyloid in the brain affected by Alzheimer's disease at a micrcoscopic scale (the size of a single neuron) and at the early stage of the disease when small amyloid fibrils are free to move and to coalesce.

In this work we perform numerical simulations on this model in order to better understand the evolution of Alzheimer's disease in the brain. The difficulty of the work rests on the method used for the time discretisation since we have a set of time dependent nonlinear coupled equations.

the report is organised as follows : ...

2 Mathematical model for the aggregation and diffusion of β-amyloid peptide

2.1 Smoluchowski equation in polymerisation

The Smoluchowski coagulation equation models various kinds of phenomena as, for example: the evolution of a system of solid or liquid particles In view of our subsequent applications, we present the appearance of the Smoluchowski equation in polymerisation.
Let P_k for $k \in N$ denote a polymer of size k : a set of k identical particles (monomers).

With time, if two polymers are very close, they are likely to merge creating a new polymer. By convention we consider only binary reactions. This phenomenon is called coalescence and we write the coalescence of a polymer of size k with polymer of size j :

$$P_k + P_j \to P_{k+j}$$

Restrictions of the model studied:
We assume that the aggregation results only from Brownian movement or diffusion (thermal coagulation). We neglect other effects such as multiple coagulation, condensation, fragmentation...
Under theses assumptions we can introduce the discrete diffusive coagulation equations :

$$\frac{\partial u_i}{\partial t}(t,x) - d_i \triangle_x u_i(t,x) = Q_i(u) \quad \text{in } [0,T] \text{ x } \Omega \tag{2.1}$$

where :

- $u_i(t,x) \geq 0$ (for $i \geq 1$) : is the variable representing the concentration of i-clusters : clusters with i identical elementary particles
- $d_i > 0$ (for $i \geq 1$) : denotes the diffusion coefficient of an i-cluster
- $Q_i(u) \quad i \geq 1$: is the coagulation term where $u = (u_i)_{i\geq 1}$, it equals :

$$Q_i(u) = Q_{g,i}(u) - Q_{l,i}(u) \tag{2.2}$$

  - where the gain term $\quad Q_{g,i} = \frac{1}{2}\sum_{j=1}^{i-1} a_{i-j,j} u_{i-j} u_j$

    describes the creation of polymers of size i by coagulation of polymers of size j and $i-j$ (clusters of size $\leq i-1$).
  - and the loss term $\quad Q_{l,i} = u_i \sum_{j=1}^{\infty} a_{i,j} u_j$

    describes the depletion (reduction) of polymers of size i after coalescence with other polymers.
  - At last, the coagulation rates $a_{i,j}$ are non nonnegative constants such that $a_{i,j} = a_{j,i}$. $a_{i,j}$ is a kinetic coefficient that represents the reaction in which an $(i+j)$-cluster is formed from an i and a j-cluster.

2.2 Mathematical model studied

2.2.1 Parameters of the microscopic model

In the mathematical model presented in ref paper the authors consider a portion of the hippocampus or of the cerebral cortex (the regions of the brain mainly affected by the

disease) whose size is comparable of the size of a neuron. With this choice of scale, it is coherent to consider that the diffusion is uniform. Moreover, it is assumed that 'large' assemblies do not aggregate with each other (which is consistent with experimental data). In the following we consider a perforated domain Ω which represents a portion of cerebral tissue where the neurons are represented by holes such that :

- $\Omega_0 \subset \mathbb{R}^3$: is a smooth bounded domain representing a portion of **cerebral tissue**
- $\overline{\Omega}_j \subset \Omega_0$ with $j = 1...N$: are regular regions representing N **neurons**
- $\overline{\Omega}_i \cap \overline{\Omega}_j = \emptyset$ if $i \neq j$: condition imposing that two neurons cannot intersect

$$\Omega := \Omega_0 \backslash \cup_{j=1}^{N} \overline{\Omega}_j$$

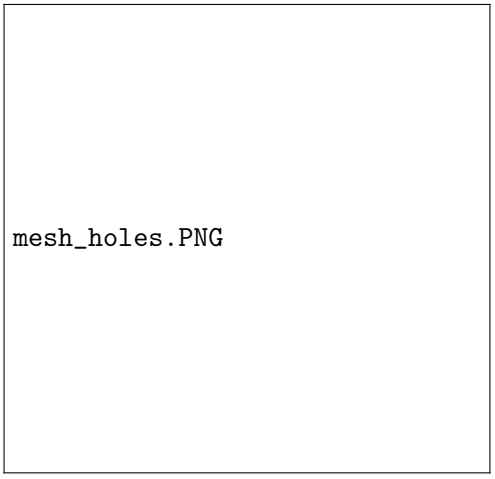

Figure 2: Representation in $\mathbb{R}^2$ for N=9 neurons

We consider a vector valued function $u = (u_1, ..., u_M$ where $M \in \mathbb{N}$ and $u_j = u_j(t, x), t \in \mathbb{R}, x \in \Omega$.

- If $1 \leq j \leq M - 1$: $u_j(t, x)$ is the molar concentration at point x and time t of an $A\beta$ assembly of j monomers.
- u_M : takes into account aggregations of more than $M - 1$ monomers ('large' assemblies)

2.2.2 The microscopic model

$A\beta$'s concentration in monomeric form

In the model we consider that $A\beta$ is produced only in monomeric form at the level of neuron membranes. This production is modeled by a non-homogeneous Neumann condition on the boundary of each neuron Ω_j : $\partial\Omega_j$ for $j = 1, ..., N$.

We also impose an homogeneous Neumann conditon on $\partial\Omega_0$ to artificially isolate the portion of cerebral tissue from its environment.
Finally at time $t = 0$, we consider that there can be an initial concentration of $A\beta$ monomers.
Thus, the following Cauchy-Neumann problem can be defined :

$$\begin{cases} \frac{\partial u_1}{\partial t}(t,x) - d_1 \triangle_x u_1(t,x) - Q_{l,1}(u) = 0 & \\ \nabla_x u_1.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_x u_1.n = \Psi_j & \text{on } \partial\Omega_j,\ j = 1...N \\ u_1(0,x) = U_1 \geq 0 & \end{cases} \tag{2.3}$$

where $0 \leq \psi_j \leq 1$ is a smooth function for $j = 1...N$ describing the production of the amyloid near the neuron membrane j . The production of $A\beta$ is not uniform over the neuronal cells, the distribution is defined by the choice of the functions ψ_j. Moreover, only neurons affected by the disease are taken into account, i.e, $\psi_j \neq 0$.
We also have $Q_1(u) = -Q_{l,1}(u) = u_1 \sum_{j=1}^{M} a_{i,j} u_j$. There is only loss since we cannot create monomers by coagulation.

$A\beta$'s concentration in assemblies

For assemblies of size $1 < m < M$,

$$\begin{cases} \frac{\partial u_m}{\partial t}(t,x) - d_m \triangle_x u_m(t,x) + Q_{l,m}(u) = Q_{g,m}(u) & \\ \nabla_x u_m.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_x u_m.n = 0 & \text{on } \partial\Omega_j,\ j = 1...N \\ u_m(0,x) = 0 & \end{cases} \tag{2.4}$$

where $Q_m(u) = Q_{g,m}(u) - Q_{l,m}(u) = \frac{1}{2} \sum_{j=1}^{i-1} a_{i-j,j} u_{i-j} u_j - u_i \sum_{j=1}^{M} a_{i,j} u_j$.

And,

$$\begin{cases} \frac{\partial u_M}{\partial t}(t,x) - d_M \triangle_x u_M(t,x) = Q_{g,M}(u) & \\ \nabla_x u_M.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_x u_M.n = 0 & \text{on } \partial\Omega_j,\ j = 1...N \\ u_M(0,x) = 0 & \end{cases} \tag{2.5}$$

where $Q_M(u) = Q_{g,M}(u) = \frac{1}{2} \sum_{\substack{j+k \geq M \\ k<M \\ j<M}} a_{j,k} u_j u_k$

The meaning of u_M differs from that of u_m, $m < M$, since it describes the sum of the densities of all the 'large' assemblies. It is assumed that large assemblies exhibit all the same coagulation properties and do not coagulate with each other. That is why the expression of the coagulation term differs from the other assemblies : there is no loss and the gain term takes also into the aggregation of assemblies $< M$ creating assemblies of size M and

greater than M.

If we consider coagulation rates of the form $a_{i,j} = \frac{\alpha}{ij}$ $\alpha > 0$, the existence of a solution is given by the following theorem obtained in ref book in article bnuro franchi :

Theorem For all $T > 0$ the Neumann-Cauchy problem (2.3), (2.4), (2.5) has a unique classical positive solution $u \in C^{1+\alpha/2,2+\alpha}([0,T]\mathrm{x}\overline{\Omega})$.

3 Numerical method

3.1 Variational formulation

Let V be a Sobolev space and $v \in V$ a test function. Let's multiply by v the coagulation diffusion equation and integrate on Ω. For all u_i with $i = 1...M$:

$$\int_\Omega \frac{\partial u_i}{\partial t} \cdot v - d_i \int_\Omega \Delta u_i \cdot v = \int_\Omega Q_i v \quad \text{on } \Omega$$

Then applying the Green theorem we have :
For $i = 1$ (a non-homogeneous Neumann condition around the neurons):

$$\int_\Omega \Delta u_1 \cdot v = -\int_\Omega \nabla u_1 \nabla v + \int_{\partial\Omega} \underbrace{(\partial_n u_1)}_{=\Psi_j \text{ on } \partial\Omega_j} v \quad \text{on } \Omega$$

$$\implies \int_\Omega \frac{\partial u_1}{\partial t} \cdot v + d_1 \int_\Omega \nabla u_1 \cdot \nabla v = \int_\Omega Q_1(u)v + \sum_{j=1}^{N} \int_{\partial\Omega_j} \Psi_j v \quad \text{on } \Omega \tag{3.1}$$

For $i > 1$, $i = 2...M$ (only homogeneous Neumann conditions) :

$$\int_\Omega \frac{\partial u_m}{\partial t} \cdot v + d_m \int_\Omega \nabla u_m \cdot \nabla v = \int_\Omega Q_m(u)v \quad \text{on } \Omega \tag{3.2}$$

We define the usual L^2-inner product $(u,v) = \int_\Omega uvdx$ $\forall u, v \in L^2(\Omega)$ and the elliptic bilinear form $a(u,v) = \int_\Omega \nabla u \cdot \nabla v$.

The variational formulation reads as follows :

$$\forall v \in V \quad \frac{\partial}{\partial t}(u_m, v) + d_m a(u_m, v) = \begin{cases} (Q_1(u), v) + \int_{\partial\Omega_{j=1...N}} \Psi_j v, & \text{if } m = 1 \\ (Q_m(u), v) & \text{otherwise} \end{cases} \tag{3.3}$$

3.2 Space discretisation

The system ref system ?? is first discretised in space by the finite element method. In this work we will consider a space Ω of dimension 2. Let T_h be a triangulation of Ω having maximal diameter h and V_h be an associated conforming finite element space. In this work, we will consider $\mathbb{P}_1$-Lagrangian finite elements. We can extract a finite element basis $(\phi_i)_{i=1..N_s}$ of V_h (with N_s the number of nodes on the mesh T_h) such that u_m for $m = 1..M$ is approximated by $u_{h,m} = \sum_{i=1}^{N_s} x_m(s_i)\phi_i$.

Space definition

In order to define V_h we first define the space of $\mathbb{P}_1$-Lagrangian functions that is a space of dimension 3 with degree 1 polynomials on ω ($\omega \subset \mathbb{R}^2$) :

$$\mathbb{P}_1(\omega) = \{p|_\omega \quad |\exists a, b, c \in \mathbb{R} \quad |\forall (x, y) \in \omega, p(x, y) = ax + by + c\}$$

We can now construct the space of $\mathbb{P}_1$-Lagrangian functions of functions locally $\mathbb{P}_1$ on each triangle. The space V_h of $\mathbb{P}_1$-Lagrangian functions associated to the triangulation T_h is defined by :

$$V_h = \{v \in C^0(\overline{\Omega}) \quad | \quad v|_K \in \mathbb{P}_1(K) \text{ for all triangle } K \in T_h\}$$

We define S_h the set of nodes of the mesh T_h :

$$S_h = \{s_i = (x_i, y_i) \quad \text{with } i \in \{1, ...N_s\}\}$$

Properties of the basis :

$$\forall i, j = 1, \dots N_s, \quad \phi_j(s_i) = \delta_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

Discretising the coagulation term

$Q(u)$ is a non linear term, it is a sum of terms t_z ($z \in \mathbb{N}$) of the form $t_z = a_{i,j}\, u_i u_j$ with $1 \leq i, j \leq M$. Replacing u_i and u_j by $u_{h,i} = \sum_k x_k \phi_k$ and $u_{h,j} = \sum_l x_l \phi_l$, we obtain :

$$t_z(x, y) = \sum_{k=1}^{N_s} \sum_{l=1}^{N_s} x_i(s_k) x_j(s_l) \phi_k(x, y) \phi_l(x, y)$$

Thanks to property (3.4) we know that on the nodes $s_y \in S_h$, the product $\phi_k(s_y)\phi_l(s_y)$ equals 0 except once when $k = l = y$ where it equals 1. As a consequence we simply have :

$$t_z(s_y) = a_{i,j} x_i(s_y) x_j(s_y) \quad \text{for all } s_y \in S_h$$

We define the projection $t_{h,z}$ of the function t_z on V_h. Using the basis $(\phi_i)_i$ we write :

$$t_{h,z}(x,y) = \sum_{y=1}^{N_s} t_z(s_y)\phi_y = \sum_{y=1}^{N_s} a_{i,j}x_i(s_y)x_j(s_y)\phi_y(x,y)$$

Therefore, to calculate the function Q(u) easily we approximate its value by its projection $Q_h(u)$ on V_h. This choice will also simplify things for the Galerkin method used on the next section.

Applying the Galerkin method

Applying a standard Galerkin procedure to (3.3) we can rewrite the variational formulation on V_h with the basis functions $(\phi_i)_{i=1...N_s}$. On V_h we use the following approximations : $v \approx v_h = \sum_{i=1}^{N_s} \phi_i, \quad u_m \approx u_{h,m} = \sum_{j=1}^{N_s} x_{m,j}\phi_j, \quad Q_m(u) \approx Q_{h,m}(u) = \sum_{y=1}^{N_s} q_{m,y}\phi_y$

such that $\forall i = 1...N_s$ and $m = 1...M$,

$$\frac{\partial}{\partial t}\sum_{j=1}^{N_s}(\phi_j,\phi_i)x_{m,j} + d_m(\phi_j,\phi_i)x_{m,j} = \begin{cases} \sum_{y=1}^{N_s}(\phi_y,\phi_i)q_{1,y} + \sum_{j=1}^{N}\int_{\partial\Omega_j}\Psi_j\phi_i, & \text{if } m=1 \\ \sum_{y=1}^{N_s}(\phi_y,\phi_i)q_{m,y} & \text{otherwise} \end{cases} \tag{3.5}$$

Or else, written in matrix form :

$$M_a\frac{\partial}{\partial t}X_m + d_m A X_m = \begin{cases} M_a Q_1 + N_1, & \text{if } m=1 \\ M_a Q_m & \text{otherwise} \end{cases} \tag{3.6}$$

with :

$$M_a = (m_{ij} = \int_\Omega \phi_i\cdot\phi_j)_{ij}, \quad A = (a_{ij} = \int_\Omega \nabla\phi_i\cdot\nabla\phi_j)_{ij}, \quad N_1 = (n1_i = \sum_{j=1}^{N}\int_{\partial\Omega_j}\Psi_j\phi_i)_i$$
$$X_m = (x_{m,i})_i, \quad Q_m = (q_{m,y})_y$$

The matrices M_a and A are respectively called the mass matrix and the stiffness matrix.

The mass lumping technique

Let's remind that we have used the finite element interpolants $Q_h(u)$ for associated to the vector of nodal values $Q(u)$ for the function for $Q(u)$. If we choose an appropriate integration quadrature formula : exact for $\mathbb{P}_1$ elements and that evaluates the integrand only on the nodes, the result will remain exact for $\mathbb{P}_1$ elements.

That having said, the mass matrix M_a is approximated using a diagonal matrix M_L called condensed or lumped matrix that is equivalent to using a trapezoidal quadrature rule on each triangle

$K \in T_h$ to approximate $m_{ij} = \sum_{K \in T_h} \int_K \phi_i \phi_j$. This rule only uses the nodal values of each triangle and thanks to the properties of finite element basis functions we obtain a diagonal matrix with the following matrix coefficients :

$$M_L = diag(\tilde{m}_{ii}) \quad \text{with} \quad \tilde{m}_{ii} = \sum_{j}^{N_s} m_{ij} \tag{3.7}$$

We notice that the diagonal elements are equal to the row sums of the original mass matrix M_a.

Using the mass lumped matrix M_L, we write :

$$M_L \frac{\partial}{\partial t} X_m + d_m A X_m = \begin{cases} M_L Q_1 + N_1, & \text{if } m = 1 \\ M_L Q_m & \text{otherwise} \end{cases} \tag{3.8}$$

3.3 Time discretisation

On the variational formulation there is a partial derivative in time. To solve the system on a bounded temporal interval, for $t \in [0, T]$, we will study different methods to discretise the equation in time. The temporal derivative is discretised by a difference quotient and the other terms by a combination of the value at time t_n and t_{n+1} depending on the method used.

We define : $u_m^n = u_m(t_n)$ for $m = 1...M$

$$\text{And} \quad \frac{\partial u_m^{n+1}}{\partial t} \approx \frac{u_m^{n+1} - u_m^n}{t_{n+1} - t_n} = \frac{u_m^{n+1} - u_m^n}{\Delta t} \quad \text{with } \Delta t = t_{n+1} - t_n$$

The equations are of the form :

$$\begin{cases} \frac{\partial u}{\partial t} = \underbrace{f(t,u)}_{Non-stiff} + \underbrace{g(t,u)}_{Stiff} \\ u(0) = u^0 \end{cases} \tag{3.9}$$

The stiff part comes from the elliptic operator and is linear whereas the other represents the coagulation which is not linear. If we use an implicit method, we end up solving a nonlinear system which is difficult to solve. On the other hand, explicit methods are often unacceptable for stiff problems due to the limited size of their stability regions.

3.3.1 Forward-Backward Euler method

Considering $Q(u)$ explicitly

On a first attempt to solve the system we will work around the difficulty of the problem by applying an implicit Euler scheme except for the coagulation term which will be expressed explicitly. At each time iteration we will define the unknown $X^{n+1} = (X_1^{n+1}, ..., X_m^{n+1}, ..., X_M^{n+1})$ solving M linear systems of the form :

$$M_L \frac{X_m^{n+1} - X_m^n}{\Delta t} + d_m A X_m^{n+1} = \begin{cases} M_L Q_1^n + N_1^{n+1}, & \text{for } m = 1 \\ M_L Q_m^n & \text{for } m = 2...M \end{cases}$$

Multiplying the equation by Δt and rearranging the terms, the scheme is written :

$$[\ M_L + \Delta t d_m A\ ]X_m^{n+1} = \begin{cases} M_L[\ X_1^n\ + \Delta t Q_1^n\ ] + \Delta t N_1^{n+1}, & \text{for } m = 1 \\ M_L[\ X_m^n\ + \Delta t Q_m^n\ ] & \text{for } m = 2...M \end{cases} \tag{3.10}$$

Stability condition on the time step

The solution $u = (u_1, ..., u_M)$ we are looking for is a vector of molar concentrations so it must be positive at all points (confirmed by the theorem of existence in section 2.2.2). In order to conserve the positivity of the solution we impose a time step such that for all $m = 1...M$:

$$\begin{aligned} \text{At each time step } t^n, \quad X_m^{n+1} \geq 0 &\Rightarrow X_m^n\ +\ \Delta t Q_m^n \quad \geq 0 \\ &\Rightarrow X_m^n\ +\ \Delta t(Q_{g,m}^n - Q_{l,m}^n) \quad \geq 0 \end{aligned}$$

We can which physically interpret this condition as the fact that we cannot loose more concentration than we have.

Neglecting $Q_{g,m}^n$ and replacing $Q_{l,m}^n$ by its expression :

$$\begin{aligned} X_m^{n+1} \geq 0 &\Rightarrow X_m^n\ -\ \Delta t X_m^n \sum_{\tilde{m}}^{M} a_{m,\tilde{m}} X_{\tilde{m}}^n \quad \geq 0 \\ &\Rightarrow X_m^n(1\ -\ \Delta t \sum_{\tilde{m}}^{M} a_{m,\tilde{m}} X_{\tilde{m}}^n) \quad \geq 0 \end{aligned}$$

This has to be true at each point so considering $X_m = (x_{m,i})_{i=1...N_s}$:

$$\begin{aligned} x_{m,i}^{n+1} \geq 0 &\Rightarrow 1 - \Delta t \sum_{\tilde{m}} a_{m,\tilde{m}} x_{\tilde{m},i}^n \leq 0 \quad \text{when } x_{m,i}^n > 0 \\ x_{m,i}^{n+1} \geq 0 &\Rightarrow \Delta t \quad \leq \frac{1}{\sum_{\tilde{m}} a_{m,\tilde{m}} x_{\tilde{m},i}^n} \end{aligned}$$

Therefore at each iteration $n = 1...It_f$, we compute a time step Δt such that :

$$\Delta t \quad \leq \min_{\substack{m,i \\ x_{m,i}^n \neq 0}} \left(\frac{1}{\sum_{\tilde{m}} a_{m,\tilde{m}} x_{\tilde{m},i}^n} \right) \tag{3.11}$$

3.3.2 Fully implicit Euler scheme

Using an approximation of $Q^{n+1}(u)$

On the first approach, we determined $Q(u)$ explicitly (at time t^n) to linearize the system for the unknown u^{n+1}. If we want to apply a fully implicit Euler scheme, we can decide to linearize the equations using an approximation of $Q_m^{n+1}(u)$ for $m = 1...M$. It would require an extra computation but will allow us to use larger time steps than in the previous method.
Let $\hat{u}$ be an available approximation of the unknown u, then we can linearize the term $Q_m(u)$ by

writing $\hat{Q}_m = Q_m(\hat{u})$ and the resulting equations are now linear. One possible option to obtain such an approximation is using a Picard iteration method.

We introduce a Picard iteration arising from the Backward Euler discretization with k as iteration counter. Let $X_m^{n,k}$ be the previously computed approximation of $\hat{X}_m$ in iteration k.
In iteration $k+1$ we want to solve :

$$[\ M_L + \Delta t d_m A\ ]X_m^{(n,k+1)} = \begin{cases} M_L[\ X_1^n\ + \Delta t Q_1^{(n,k)}\ ] + \Delta t N_1^{(n,k+1)}, & \text{for } m = 1 \\ M_L[\ X_m^n\ + \Delta t Q_m^{(n,k)}\ ] & \text{for } m = 2...M \end{cases} \tag{3.12}$$

We start with $Q_m^{(n,0)} = Q_m^n$ and the final converged value $\hat{Q}_m = Q_m^{n,k}$ ($\hat{X}_m = X_m^{n,k}$) for sufficiently large k.

Stopping criteria : The iteration method is terminated when the change in the solution is smaller than a tolerance ϵ :

$$\max_m \left\| \frac{X_m^{(n,k+1)} - X_m^{(n,k)}}{X_m^{(n,k)}} \right\|_{L^2(\Omega)}^2 \leq \epsilon \tag{3.13}$$

// TODO
In the Picard iterations we do not care about the previously presented condition on the time step -> even if we have negative values for X at the beginning it is suppose to converge to a good approximation. NEED help formalising
//

To compute X^{n+1}, we solve the following system with the approximated values $\hat{Q}_m$ computed by the Picard iteration method :

$$[\ M_L + \Delta t d_m A\ ]X_m^{n+1} = \begin{cases} M_L[\ X_1^n\ + \Delta t \hat{Q}_1\ ] + \Delta t N_1^{n+1}, & \text{for } m = 1 \\ M_L[\ X_m^n\ + \Delta t \hat{Q}_m\ ] & \text{for } m = 2...M \end{cases} \tag{3.14}$$

3.3.3 Crank-Nicolson scheme

In order to increase the accuracy in time we can use a Crank-Nicolson scheme. The system to solve is the following :

$$[\ M_L + \frac{\Delta t}{2} d_m A\ ]X_m^{n+1} = \begin{cases} M_L[\ X_1^n\ + \frac{\Delta t}{2}(Q_1^{n+1} + Q_1^n)\ ] - \frac{\Delta t}{2} d_m A X_1^n + \Delta t N_1^{n+1/2}, & \text{for } m = 1 \\ M_L[\ X_m^n\ + + \frac{\Delta t}{2}(Q_m^{n+1} + Q_m^n)\ ] - \frac{\Delta t}{2} d_m A X_m^n & \text{otherwise} \end{cases} \tag{3.15}$$

As on section 3.3.2, we apply the Picard iteration method to linearize the system approximating Q_m^{n+1}. Hopefully by iterating until some convergence criterion was met, one would preserve the good stability properties of this method. We write :

$$[\ M_L + \frac{\Delta t}{2} d_m A\ ]X_m^{n+1} = \begin{cases} M_L[\ X_1^n\ + \frac{\Delta t}{2}(\hat{Q}_1 + Q_1^n)\ ] - \frac{\Delta t}{2} d_m A X_1^n + \Delta t N_1^{n+1/2}, & \text{for } m = 1 \\ M_L[\ X_m^n\ + + \frac{\Delta t}{2}(\hat{Q}_m + Q_m^n)\ ] - \frac{\Delta t}{2} d_m A X_m^n & \text{otherwise} \end{cases} \tag{3.16}$$

with ... (not done)

3.3.4 IMEX Runge-Kutta method

Another solution to increase the order of accuracy in time is to use an implicit-explicit Runge-Kutta (IMEX-RK) method. First we introduce the general formulation of the scheme on a system of equations of the form (3.9). A general IMEX-RK scheme applied to the differential system

$$u' = f(u) + g(u), \tag{3.17}$$

reads

$$U^{(i)} = u^n + \Delta t \sum_{j=1}^{i-1} \tilde{a}_{ij} f(U^{(j)}) + \Delta t \sum_{j=1}^{\nu} a_{ij} g(U^{(j)}) \tag{3.18}$$

$$u^{n+1} = u^n + \Delta t \sum_{i=1}^{\nu} \tilde{w}_i f(U^{(i)}) + \Delta t \sum_{i=1}^{\nu} w_i g(U^{(i)}). \tag{3.19}$$

The matrices $\tilde{A} = (\tilde{a}_{ij})$, $\tilde{a}_{ij} = 0$ for $j \geq i$ and $A = (a_{ij})$ are $\nu \times \nu$ matrices.
Considering $\tilde{a}_{ij} = 0$ for $j \geq i$ guarantees that f is always evaluated explicitly. Choosing $a_{ij} = 0$, for $j > i$ results in a diagonally implicit (DIRK) scheme for g. We also define the vectors c and $\tilde{c}$ such that $c_i = \tilde{c}_i = \sum_{j=1}^{\nu} a_{ij} = \sum_{j=1}^{\nu} \tilde{a}_{ij}$.

Here, u^n is an approximate value of $u(t_n)$ and $U^{(i)}$ are intermediate variables at times $t_{n,i} = t_n + c_i \Delta t$. Therefore, we read that in carrying out a step we evaluate ν stage values $U^{(1)}, ..., U^{(\nu)}$ to compute u^{n+1}. In general, an IMEX Runge–Kutta scheme, is characterised by the above defined two matrices and the coefficient vectors $\tilde{w} = (\tilde{w}_1, ..., \tilde{w}_\nu)^T$, $w = (w_1, ..., w_\nu)^T$.

This type of scheme can be represented by a double tableau in the usual Butcher notation,

$$\begin{array}{c|c} \tilde{c} & \tilde{A} \\ \hline & \tilde{w} \end{array} \qquad\qquad \begin{array}{c|c} c & A \\ \hline & w \end{array} \tag{3.20}$$

// TODO
Remarks The Forward-BAckward EUler method is actually an IMEX-RK scheme of order 1. Write correspondind double tableau.
//

In this report we study a method denoted $ARS(2,2,2)$ which is second order accurate. Using the notation above (3.20), it is written,

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \gamma & \gamma & 0 & 0 \\ 1 & \delta & 1-\delta & 0 \\ \hline & \delta & 1-\delta & 0 \end{array} \qquad \begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \gamma & 0 & \gamma & 0 \\ 1 & 0 & 1-\gamma & \gamma \\ \hline & 0 & 1-\gamma & \gamma \end{array} \qquad \begin{array}{l} \gamma = 1 - \frac{\sqrt{2}}{2} \\ \delta = 1 - \frac{1}{2\gamma} \end{array} \tag{3.21}$$

Figure 3: Tableau for the explicit (left) implicit (right) IMEX-ARS(2,2,2) scheme

This scheme requires $\nu = 3$ stages, applying the general formula (3.18)-(3.19), it is written :

$$U^{(1)} = u^n \tag{3.22}$$

$$U^{(2)} = u^n + \Delta t \gamma f(U^{(1)}) + \gamma \Delta t g(U^{(2)}) \tag{3.23}$$

$$u^{n+1} = U^{(3)} = u^n + \Delta t[\delta f(U^{(1)}) + (1-\delta) f(U^{(2)})] + \Delta t[(1-\gamma) g(U^{(2)}) + \gamma g(U^{(3)})] \tag{3.24}$$

We have $u^{n+1} = U^{(3)}$ because the 3^{rd} and last line of the double tableau (3.21) are equal. We also see that since the first row is equal to 0, $U^{(1)} = u^n$ so this scheme actually has only one intermediate state to compute u^{n+1}.

Applying this method to our system

In our model f represents the elliptic operator (diffusion term) and g the coagulation function. Applying this time discretization to the system (3.8) we have :

For $m = 2...M$,

$$\begin{aligned}
X_m^{(1)} &= X_m^n \\
[\, M_L + \gamma \Delta t d_m A \,] X_m^{(2)} &= M_L [\, X_m^n + \gamma \Delta t Q_m(X^{(1)})] \\
[\, M_L + \gamma \Delta t d_m A \,] X_m^{(3)} &= M_L [\, X_m^n + \Delta t[\delta Q_m(X^{(1)}) + (1-\delta) Q_m(X^{(2)})] \,] \\
&\quad - \Delta t (1-\gamma) d_m A X_m^{(2)}
\end{aligned} \tag{3.25}$$

For the concentration in monomers we have an additional term N^1 coming from the Neumann condition on the neuron membranes. For simplification we consider it constant in time and have :

For m=1,

$$\begin{aligned}
X_1^{(1)} &= X_1^n \\
[\, M_L + \gamma \Delta t d_1 A \,] X_1^{(2)} &= M_L [\, X_1^n + \gamma \Delta t Q_1(X^{(1)})] + \gamma \Delta t N^1 \\
[\, M_L + \gamma \Delta t d_1 A \,] X_1^{(3)} &= M_L [\, X_1^n + \Delta t[\delta Q_1(X^{(1)}) + (1-\delta) Q_1(X^{(2)})] \,] \\
&\quad - (1-\gamma) \Delta t d_1 A X_1^{(2)} + \Delta t N^1
\end{aligned} \tag{3.26}$$

Stability condition on the time step

As for the Forward-Backward Euler method in section 3.3.1 the coagulation term is expressed explicitly and we need to impose a similar condition on the time step to conserve the positivity of the solution. The difficulty here is that we can define a time step $\Delta t^{(1)}$ to compute $X_m^{(2)}$ but then we need to verify that this time step is small enough to compute $X_m^{(3)}$.

Following the same reasoning as in 3.3.1 we define a condition to compute $X_m^{(2)}$:

$$\begin{aligned}
\text{At each time step } t^n, \quad X_m^{(2)} \geq 0 &\Rightarrow X_m^n \; + \; \gamma \Delta t Q_m(X^{(1)}) \quad \geq 0 \\
&\Rightarrow X_m^{(1)} \; + \; \gamma \Delta t Q_m(X^{(1)}) \quad \geq 0 \quad \text{since } X_m^n = X_m^{(1)}
\end{aligned}$$

The only difference with the Forward-Backward Euler scheme is the parameter γ. Therefore with a simple change of variables, the condition on Δt at each time iteration is :

$$\gamma \Delta t \quad \leq \min_{\substack{m,i \\ x_{m,i}^{(1)} \neq 0}} \left(\frac{1}{\sum_{\tilde{m}} a_{m,\tilde{m}} x_{\tilde{m},i}^{(1)}} \right) \tag{3.27}$$

The condition on Δt to compute $X_m^{(3)}$ is defined using a similar reasoning and making some simplifications that seem to be coherent with the numerical tests performed.

$$\text{At each time step } t^n > 0, \quad X_m^{(3)} \geq 0 \Rightarrow \; X_m^n + \Delta t[\delta Q_m(X^{(1)}) + (1-\delta)Q_m(X^{(2)})] \quad \geq 0$$

Neglecting the gain term $Q_{g,m}$ in $Q_m(X^{(1)})$ and $Q_m(X^{(2)})$:

$$X_m^{(3)} \geq 0 \Rightarrow \; X_m^n - \; \Delta t[\delta Q_{l,m}(X^{(1)}) + (1-\delta)Q_{l,m}(X^{(2)})] \quad \geq 0$$

Unlike in the precedent case, we have a term evaluated in $X^{(2)}$ so we cannot factor by $X_m^{(1)}$ i.e X_m^n. Then, directly considering $X_m = (x_{m,i})_i$:

$$\text{when } x_{m,i}^n > 0 \quad x_{m,i}^{(3)} \geq 0 \Rightarrow \Delta t \quad \leq \frac{x_{m,i}^n}{[\ \delta Q_{l,m}(x_i^{(1)}) + (1-\delta)Q_{l,m}(x_i^{(2)})\ ]}$$

with $x_i^{(.)} = (x_{m,i}^{(.)})_m$

Finally at each iteration $n = 1...It_f$, the condition on the time step Δt is :

$$\Delta t \quad \leq \min_{\substack{m,i \\ x_{m,i}^n \neq 0}} \left(\frac{x_{m,i}^n}{[\ \delta Q_{l,m}(x_i^{(1)}) + (1-\delta)Q_{l,m}(x_i^{(2)})\ ]} \right) \tag{3.28}$$

// TODO
Consider case when $x_{m,i}^n = 0$ but $Q_{l,m}(x_i^{(2)}) > 0$, ??? What to do
If we impose $X^{(2)} - \Delta t^{(3)} Q(X^{(2)} > 0$? <- physical condition instead of from scheme to compute $X^{(3)}$
//

Let's define $\Delta t^{(2)}$ the time step used to compute $X^{(2)}$ and $\Delta t^{(3)}$ the larger time step respecting the stability condition (3.28). If $\Delta t^{(3)} < \Delta t^{(2)}$ the stability condition is not respected. Therefore, we need to recompute $X^{(2)}$ with a smaller $\Delta t^{(2)}$ and recheck the condition (3.28) with the $\Delta t^{(3)}$ associated to the new value of $X^{(2)}$. At each iteration, the algorithm is as follows :

Algorithm 1 Calculate $X^{(3)}$ i.e X^{n+1}

Require: $0 \leq k \leq 1$ parameter to choose smaller $\Delta t^{(2)}$
Ensure: X^{n+1} computed respecting stability condition
  $X^{(1)} \leftarrow X^n$
  Calculate $\Delta t^{(2)}$ respecting (3.27)
  Compute $X^{(2)}$
  Calculate $\Delta t^{(3)}$ respecting (3.28)
  while $\Delta t^{(3)} < \Delta t^{(2)}$ **do**
    $\Delta t^{(2)} \leftarrow k\ \Delta t^{(2)}$
    $X^{(2)} \leftarrow$ computed with new $\Delta t^{(2)}$
    $\Delta t^{(3)} \leftarrow$ computed with new $X^{(2)}$
  end while
  Compute $X^{(3)}$
  return X^{n+1}

4 Convergence analysis

In this part we study the convergence of the different schemes mentioned above. We present the results of the convergence tests performing some sample computations to verify the convergence with respect to the mesh size (Δx) and the time step (Δt). The theoretical convergence rates of the methods studied are known, so verifying the order of convergence of a scheme is a good check to show that it is correctly implemented.

The variable parameters for the convergence speed are the number of grid points used to perform the simulation. We distinguish the accuracy in space and the accuracy in time. The convergence in space depends on the number of nodes N_s which is inversely proportional to the characteristic mesh size h. The convergence in time depends on the number of iterations performed It_f ($= T/\Delta t$ for Δt constant).

Considering the variable parameter h, a numerical approximation u_h is said to converge to its exact value u with order p if there exists a number C independent of h such that:

$$|u_h - u| \leq Ch^p \tag{4.1}$$

at least for suficiently small h. This is written as $|u_h - u| = Ch^p + O(h^{p+1}) = O(h^p)$ using big O notation.
To determine the order p of a given sequence of approximations $u_{h_1}, u_{h_2}, ...$ we can either be in the situation that the exact value u is known, or, in our case, that u is unknown.

Here, we use a discretization/integration both in time and space. We say that our scheme is of order p in space and p' in time when we have :

$$|u_h - u| = \mathcal{O}(\Delta x^p) + \mathcal{O}(\Delta t^{p'}) \tag{4.2}$$

We will on a first part test the order of accuracy of our space discretization and on a second part the convergence rates for our different time discretization methods. Finally we will study the full space-time convergence.

4.1 Space convergence

To discretize the space we used $\mathbb{P}_1$ finite elements. This method is of order 2 considering the $||.||_{L^2(\Omega)}$ norm. For the error in space we have :

$$\begin{aligned} ||u_h - u||_{L^2(\Omega)} &\leq C_1 h^2 ||u||_{H^2(\Omega)} \\ &\leq Ch^2 \end{aligned} \tag{4.3}$$

with $C = C1||u||_{H^2(\Omega)}$ independent of h.
And $||u_h - u||_{L^2(\Omega)} = \sqrt{\int_\Omega (u_h - u)^2 d\Omega}$

4.1.1 Test : Laplacian equation

In order to check the convergence rate in space we consider on a first part the Laplacian equation which does not depend of time :

$$\begin{cases} -\triangle u &= 0 & \text{on } \Omega = [0,1]^2 \\ u &= u(x,y) & \text{on } \partial\Omega \end{cases} \tag{4.4}$$

We consider the function $u(x,y) = x^3 - 3xy^2$ which is a solution of the above problem. Now that we have the expression of an exact solution we can compute the error sequence $(||u - u_h||_{L^2})_h$. In order to get a precise number p we will look at the ratios $u - u_h$ and $u - u_{h/2}$,

$$\frac{||u - u_h||_{L^2}}{||u - u_{h/2}||_{L^2}} = \frac{Ch^p + \mathcal{O}(h^{p+1})}{C(h/2)h^p + \mathcal{O}((h/2)^{p+1})} = 2^p + \mathcal{O}(h). \tag{4.5}$$

Hence,

$$log_2 \left(\frac{||u - u_h||_{L^2}}{||u - u_{h/2}||_{L^2}} \right) = p + \mathcal{O}(h). \tag{4.6}$$

Computing the error for $h = 0.2, ..., 0.0125$, the corresponding convergence table is :

h	$\|\|u_h - u\|\|_{L^2}$	P
0.2	1.0E-2	-
0.1	2.5E-3	2
0.05	6.25E-4	2
0.025	1.5625E-4	2
0.0125	3.906E-5	2

Table 1: Convergence table

with $P = log_2 \left(\frac{||u_h - u||_{L^2}}{||u_{h/2} - u||_{L^2}} \right)$.

4.1.2 Test : Coagulation equation

The coagulation problem without the diffusion operator is written :

$$\begin{cases} \frac{\partial u_1}{\partial t}(t,x,y) - Q_{l,1}(u) = 0 & \\ \nabla_{x,y} u_1.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_{x,y} u_1.n = \Psi_j & \text{on } \partial\Omega_j,\ j = 1...N \\ u_1(0,x,y) = U_1(x,y) \geq 0 & \end{cases} \tag{4.7}$$

For assemblies of size $1 < m < M$,

$$\begin{cases} \frac{\partial u_m}{\partial t}(t,x,y) + Q_{l,m}(u) = Q_{g,m}(u) & \\ \nabla_{x,y} u_m.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_{x,y} u_m.n = 0 & \text{on } \partial\Omega_j,\ j = 1...N \\ u_m(0,x,y) = 0 & \end{cases} \tag{4.8}$$

And,

$$\begin{cases} \frac{\partial u_M}{\partial t}(t,x,y) = Q_{g,M}(u) & \\ \nabla_{x,y} u_M.n = 0 & \text{on } \partial\Omega_0 \\ \nabla_{x,y} u_M.n = 0 & \text{on } \partial\Omega_j,\ j = 1...N \\ u_M(0,x,y) = 0 & \end{cases} \tag{4.9}$$

Do not know how to explain it We can observe that the distribution in space depends only on the initial concentration in monomers $U_1(x,y)$ and the production function $\psi_j(x,y)$. Let us remember that we consider $U_1 = 0$ and $\psi_0 = 0.5$ a constant. Therefore, the resulting concentrations will be equal to 0 on all Ω except around the neuron membrane.

4.2 Time convergence

4.2.1 Simple heat equation

Convergence analysis for the Crank Nicolson scheme

The Crank Nicholson scheme is of order 2 in time and space, hence :

$$||u_h - u||_{L^2} = O(\Delta x^2) + O(\Delta t^2)$$

We will verify the convergence rate of the scheme for two diffusion problems ; one for which u is known and one more similar to our problem for which u is unknown.

Test case : with exact solution

Firstly, we study the convergence of the scheme on a diffusion problem (4.12) with Dirichlet conditions on the border and for which we already know the solution :

$$\begin{cases} \frac{\partial u}{\partial t} - \triangle u = f & \text{on } \Omega = [0,1] \times [-0.5, 0.5], t > 0 \\ u(t,x,y) = 0 & \text{on } \partial\Omega, t > 0 \\ u(t=0) = u_0(x,y) & \text{on } \Omega \end{cases} \tag{4.10}$$

At each iteration we solve a linear system of the form :

$$[\ M_L + \frac{\Delta t}{2} A\ ]X^{n+1} \ = \ [M_L - \frac{\Delta t}{2} A\ ]X^n + \frac{\Delta t}{2}(F^{n+1} + F^n) \tag{4.11}$$

with $F^n = (f_i^n)_i = (\int_\Omega f(t_n, x, y)\phi_i)_i$

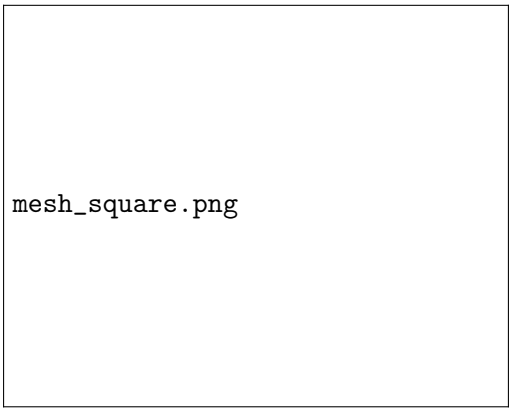

Figure 4: Discretization of Ω with $\Delta x = 0.05$

We use $u = u(t, x, y) = 5cos(10t)sin(2\pi x)cos(\pi y)$ which equals 0 on $\partial\Omega$ and choose f such that :

$$f(t, x, y) = \frac{\partial u}{\partial t} - \triangle u \quad \text{with} \quad \begin{array}{l} \frac{\partial u}{\partial t} = -50sin(10t)sin(2\pi x)cos(\pi y) \\ \frac{\partial^2 u}{\partial x^2} = -4\pi^2 u(t, x, y) \\ \frac{\partial^2 u}{\partial y^2} = -\pi^2 u(t, x, y) \end{array}$$

$$\implies \quad f(t, x, y) = -50sin(10t)sin(2\pi x)cos(\pi y) + 5\pi^2 u(t, x, y)$$

Taking equal time step and space size h we should have :

$$||u_h - u||_{L^2} = O(\Delta x^2) + O(\Delta t^2) = O(h^2)$$

On the convergence table we have the results of :

- $||u_h(T) - u(T)||_{L^2(\Omega)}$: the L^2 error at time $T = 3$,
- $L^\infty(0, T; L^2(\Omega))$: the maximum L^2 error for $t \in [0, T]$,
- $P_T = log_2\left(\frac{||u_h(T)-u(T)||_{L^2}}{||u_{h/2}(T)-u(T)||_{L^2}}\right)$ and $P = log_2\left(\frac{\max\limits_t(\ ||u_h(t)-u(t)||_{L^2}\ )}{\max\limits_t(\ ||u_{h/2}(t)-u(t)||_{L^2}\ )}\right)$: the respective convergence rates.

$\Delta t = \Delta x$	$L^2(T;\Omega)$	P_T	$L^\infty(0,T;L^2(\Omega))$	P
0.2	2.9334E-1	-	3.9007E-1	-
0.1	4.9635E-2	2.56	1.0161E-1	1.94
0.05	1.1141E-2	2.12	2.0800E-2	2.29
0.025	2.7950E-3	2.03	5.0023E-3	2.06
0.0125	6.9524E-4	2.01	1.2506E-3	1.99

Table 2: Convergence table

Remarks The errors are computed considering the projection of the exact solution on V_h.

Test case : with unknown u

In this test case, we study a diffusion problem on a perforated domain with Neumann conditions on the borders. We are considering a problem with similar conditions as the original problem for the concentration in monomers. The domain considered is $\Omega = \Omega_0 \backslash \Omega_1$ with $\Omega_0 = [0,1]^2$ and $\Omega_1 = \mathcal{C}((0.5, 0.5), R)$ a circle of radius R representing one centered neuron.

$$\begin{cases} \frac{\partial u}{\partial t} - d \triangle u & = 0 & \text{on } \Omega, t > 0 \\ u(t,x,y) & = 0 & \text{on } \partial\Omega_0, t > 0 \\ u(t,x,y) & = \psi_1 & \text{on } \partial\Omega_1, t > 0 \\ u(t=0) & = U1(x,y) & \text{on } \Omega \end{cases} \tag{4.12}$$

For the convergence analysis we define the parameters :

- $R = 0.2$, circle's radius
- $d = 1$, diffusion coefficient
- $\psi_1 = 0.5$, constant production function around the neuron
- $U1 = 0$, initial concentration on Ω

At each time iteration we solve a linear system of the form :

$$[\ M_L + \frac{\Delta t}{2} dA\ ] X^{n+1} \ = \ [M_L - \frac{\Delta t}{2} dA\ ] X^n + \Delta t N^1 \tag{4.13}$$

with $N^1 = (n_i^n)_i = (\int_{\partial\Omega_1} \psi_1 \phi_i)_i$ (ψ_1 is constant in time and space).

With u unknown there are two main approaches. The first one is to compute a reference solution with a very small Δt and Δx and then proceed as in the case of u known. This strategy is quite expensive since u_h is costly to compute. We can use the second approach which consists at looking at ratios of differences between u_h computed for different h. For $h = \Delta x = \Delta t$ we get,

$$\frac{||u_h - u_{h/2}||_{L^2}}{||u_{h/2} - u_{h/4}||_{L^2}} = \frac{Ch^p - C(h/2)^p + \mathcal{O}(h^{p+1})}{C(h/2)h^p - C(h/4)^p + \mathcal{O}((h)^{p+1})} = 2^p + \mathcal{O}(h). \tag{4.14}$$

Hence,

$$log_2 \left(\frac{||u_h - u_{h/2}||_{L^2}}{||u_{h/2} - u_{h/4}||_{L^2}} \right) = p + \mathcal{O}(h). \tag{4.15}$$

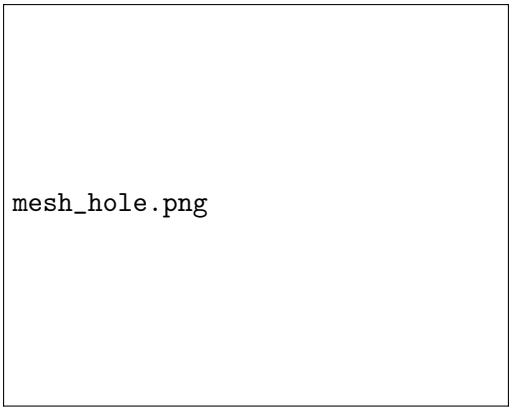

Figure 5: Discretization of Ω with characteristic element size $\Delta x = 0.05$

$\Delta t = \Delta x$	$\|\|u_h - u_{h/2}\|\|_{L^2}$	P
0.2	1.463E-1	1.90
0.1	3.924E-2	1.97
0.05	9.881E-3	1.99
0.025	2.472E-3	-
0.0125	-	-

Table 3: Convergence table for $T = 3$

with $P = log_2\left(\frac{||u_h - u_{h/2}||_{L^2}}{||u_{h/2} - u_{h/4}||_{L^2}}\right)$.

Remarks To compute P for a given h the errors are evaluated considering the same triangular mesh T_h with characteristic element size h.

4.2.2 Simple coagulation

Test case : Explicit Euler scheme

//order 1 ok
tests performed for M=3 Tf=2 dt=0.04 to 0.0025 and h fixed=0.05 ($h^2 = 0.0025$) //
//

Test case : Implicit Euler scheme (Picard approximation)

order 1 ok for v2
tests performed for M=3 Tf=4 dt=0.2 to 0.0125 and h fixed=0.05 ($h^2 = 0.0025$)

//
//
with new domain
tf=2; n=160; eps=0.2

dt1=0.0025

err1(1)= 1.96333e-05 err2(1)= 9.64028e-06 err3(1)= 4.77991e-06 err1(2)= 7.37707e-06 err2(2)= 3.52091e-06 err3(2)= 1.72235e-06 err1(3)= 0.000420664 err2(3)= 0.000207825 err3(3)= 0.00010334
p1(1)= 1.02616; p1(2)= 1.0671; p1(3)= 1.0173;
p2(1)= 1.01209; p2(2)= 1.03157; p2(3)= 1.00797;

dt1=0.005

err1(1)= 4.09103e-05 err2(1)= 1.96333e-05 err1(2)= 1.63243e-05 err2(2)= 7.37707e-06 err1(3)= 0.00086465 err2(3)= 0.000420664
p1(1)= 1.05916; p1(2)= 1.1459; p1(3)= 1.03945;

dt1=0.01

err1(1)= 8.97635e-05 err2(1)= 4.09103e-05 err1(2)= 4.02848e-05 err2(2)= 1.63243e-05 err1(3)= 0.0018418 err2(3)= 0.00086465
p1(1)= 1.13366; p1(2)= 1.30321; p1(3)= 1.09093;

dt1=0.02

err1(1)= 0.000211992 err2(1)= 8.97635e-05 err1(2)= 0.000111946 err2(2)= 4.02848e-05 err1(3)= 0.00414322 err2(3)= 0.0018418
p1(1)= 1.23981; p1(2)= 1.47449; p1(3)= 1.16964;

dt1=0.04

err1(1)= 0.000484266 err2(1)= 0.000211992 err1(2)= 0.000281832 err2(2)= 0.000111946 err1(3)= 0.00914712 err2(3)= 0.00414322
p1(1)= 1.19179; p1(2)= 1.33203; p1(3)= 1.14256;

Avec 0.08 pas bien !

change eps=0.1; Tf=8

dt1=0.4 p=1

dt1=0.2 p=1

dt1=0.1 p=0.8

dt1=0.05 p=1.1

dt1=0.025 p=1.05

Test case : Crank Nicolson (Picard approximation)

4.2.3 Full model

t1 t2 t3 = 2 2 2 2 err1(1)= 0.00113428 err2(1)= 0.000237392 err3(1)= 0.000126105 p1(1)= 2.25643; p2(1)= 0.912649; err1(2)= 0.00104102 err2(2)= 0.000494246 err3(2)= 0.000252663 p1(2)= 1.07469; p2(2)= 0.968015; err1(3)= 0.00179926 err2(3)= 0.00103676 err3(3)= 0.000605679 p1(3)= 0.795331; p2(3)= 0.775451;

5 Numerical results

5.1 Simple heat equation in 2D

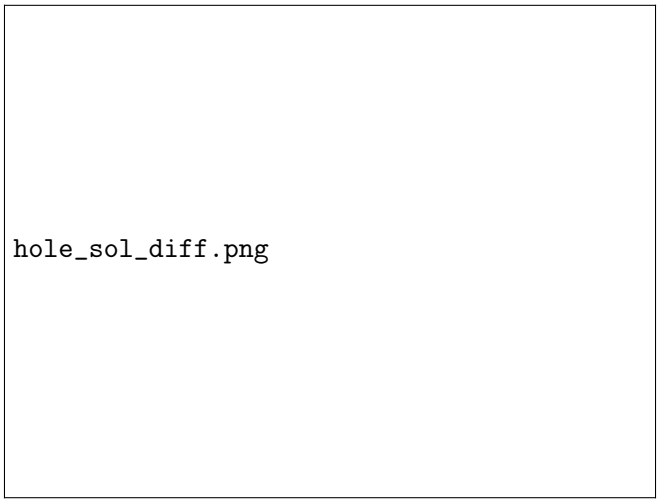

Figure 6: Solution at $T = 3$ and $h = 0.05$ for problem (4.12)

5.2 Full model 2D with holes

6 Conclusion

Part II

Macroscopic model for the aggregation and diffusion of β-amyloid peptide