POLITECNICO DI MILANO

Scuola di Ingegneria Industriale e dell'Informazione

Corso di Laurea Magistrale in Ingegneria Matematica

Brain Tumors: mathematical modeling and numerical simulations for cancer growth and invasion

Relatori:

Prof. Davide Ambrosi

Prof. Sandro Salsa

Elaborato di laurea di:

Matteo Manica

Matr. 766089

Anno Accademico 2012-2013

Contents

List of Figures

List of Tables

Aknowledgements

I'd like to thank all the people that has helped me during my thesis work.
First of all I'd like to thank my advisors, Prof. Sandro Salsa and Prof. Davide Ambrosi, for their precious guide and their help during the all the phases of this paper's development. Their passion and knowledge in the field of applied math gave me a capital support.
I'd like to thank also Prof. Luca Mainardi for the help given during the phase of images segmentation and for the opportunity that he gave me to get in contact with *INT* (*Istituto Nazionale dei Tumori*).
Special thanks goes to Dr. Paolo Potepan, from *INT*, for his availability and his professionalism, he gave me a great help in understanding the processes involved in brain's tumor growth, and with the interpretation of the *MRI*s.
I want to say thank to Dr. Eros Montin for his great help during the process of images segmentation.
I'd like to thank also Dr. Elena Faggiano, she helped me a lot with the software used for the numerical set up and the images processing, her availability gave me a capital help during all the thesis work.

Sommario

Negli ultimi decenni l'utilizzo di modelli matematici per descrivere l'evoluzione dei tumori si è incredibilmente sviluppato. In questo lavoro l'attenzione è concentrata in particolar modo sui modelli diffusione–reazione, specialmente nel modellizzare la generazione e lo sviluppo dei tumori cerebrali infantili. Questi modelli sono stati modificati e sviluppati nel corso degli anni, grazie al lavoro di Murray, Swanson et al., che hanno largamente studiato il fenomeno biologico al fine di ottenere una descrizione matematica soddisfacente.

Lo scopo di questo lavoro è di valutare la precisione di questo tipo di modelli mediante simulazioni numeriche di un codice elementi finiti, su una griglia derivata da una serie di risonanze magnetiche effettuate su di un paziente test. L'analisi consiste nello studiare e comparare, dal punto di vista analitico e sperimentale, i risultati ottenuti tramite un modello a diffusione lineare (tipico in letteratura), similare ad un equazione del calore, ed un modello a diffusione non lineare proposto nel presente elaborato, basato sull'equazione dei mezzi porosi.

Abstract

In the last decades the use of mathematical models to describe the evolution of a cancer has been greatly developed. In this work the attention is focused on diffusion–reaction models, especially in generation and development of childhood brain cancers. These models has been modified and improved during the years, thanking the work of Murray, Swanson et al., that have widely and intensively studied the biological phenomenon in order to obtain a satisfactory mathematical description.

The aim is to evaluate the precision of this kind of models through simulations of a finite elements code on a mesh derived from MRIs of a test patient. The whole analysis consists in studying and comparing, both under an analytical and a numerical point of view, the results obtained through a usual linear diffusion model, similar to the heat equation, and a non linear diffusion model implemented in this paper, based on the porous medium equation.

Keywords

FE, *Galerkin*, MRI, *Heat Equation*, *Porous Medium Equation*, *Tumor*, *Growth Diffusion Reaction Models*, *Mesh Generation*, *VTK*, *FreeFem++*.

Introduction

Mathematical Biology has encountered a huge development in the last decades, and in the last years interactions between mathematics and medicine have been even more significant and intense. Especially the use of mathematical models to predict and describe complex physical and biological phenomena has incredibly spread in the scientific community. Modeling tumor's growth is an interesting and intricate quest and there are a lot of ways to study this kind of processes. The approach depends strictly on which kind of tumor is considered and where the cancer is located.

Concerning in particular brain tumors, the details of their spatio–temporal growth are still not well understood. Indeed their development and expansion process present interesting challenges from the mathematical point of view. An attempt to describe the growth of this kind of cancers can be done using a diffusion–reaction system of partial differential equations, as can be seen in [Mur03] where an heat equation is used.

In this kind of models the tumor is studied by analyzing the diffusion of its cells or its cellular concentration in the brain. The generation of new cancerous cells has been simulated via an appropriate reaction term, and the movement of the cells in brain tissues by a proper diffusion term, that takes into consideration spatial heterogeneity by varying the diffusion coefficient over the relevant domain.

These models are easy to implement and study. While they work well in some test cases, they present some problems dealing with more complex problems. The determination of the diffusive parameters is a complex matter and a lot of articles have been devoted to it, see for example [JMD$^+$05]. This matter is somewhat critical, indeed the brain is characterized by a complicated geometry,to describe accurately its structure and the heterogeneity of the tissues is mandatory to create a consistent mathematical model. Another lack of these models is the linearity in the diffusion term, that could be not suitable to capture the propagation front of the main cellular mass of the tumor, not representing the degeneration of the invaded tissues.

Normally the tumors studied via these models are gliomas and similar brain tumors like glioblastoma. In this work the focus is on the medulloblastoma, a highly malignant primary brain tumor, discovered in 1925, that usually originates in the cerebellum or posterior fossa that commonly affects the cerebra of children. For this type of cancer there is no much literature that describes the study of its growth using diffusion–reaction models.

In this work is presented an analysis of the growth of such a tumor on the basis of the previously mentioned models, studying their properties and examining critically their capability to describe the phenomenon. Furthermore is proposed a non linear version of the model, based on the porous medium equation, where a non linearity in the diffusion term, which is supposed to be a function of the tumor's cells concentration, is introduced. By this modification of the linear model. it is possible to describe the degeneration of invaded tissues and obtain a solution that captures the front of propagation of the expanding cancer. The modified model is interesting for another reason. The classic linear model tend to overestimate the size of the growing cancer, while non linear diffusion alters the speed of cellular propagation and alleviates the over diffusive effects. That means that a sharper estimation of the cancer's volume is found. This aspect can be crucial when evaluating brain's invasion during tumor's growth and makes the model a more helpful instrument from a medical point of view. Indeed using it to predict the growth in time of a cancerous mass might help to understand how the disease will affect the patient and could suggest when and where to intervene surgically. The present work describes the two models above mentioned and analyzes their effect on a real case of growing medulloblastoma, discussing their own properties and comparing the results obtained through numerical simulation. Chapter 1 contains an overview on the history of this kind of models, the derivation of the governing equations from the phenomenon, and a study upon the existence of a unique solution in linear and non linear diffusion case. The attention is focused on the correct choice of the parameters; in particular, concerning the non linear diffusion model, an analysis on the growth observed on the MRIs (*Magnetic Resonance Images*) is carried on, in order to select the correct function of the cell concentration to be introduced in the diffusion term.

In Chapter 2 are briefly presented the discretization adopted for both the problems and the numerical methods.

The tests are performed over a brain mesh obtained from MRIs provided by *"Banca Dati www.tumori.net"* of INT (*Istituto Nazionale dei Tumori*). The mesh analysis and its creation derived from MRIs is discussed in Chapter 3, where the techniques used to generate the computational mesh of the brain are introduced and the main issues about obtaining a good representation of the biological domain without increasing too much the computational costs to implement the simulations are discussed. Two short paragraphs, one about the segmentation method used to analyze the MRIs and the other one concerning the software adopted for the simulations, are included.

In Chapter 4 are presented the main results obtained about the capability of these models to describe the growth of a brain tumor, especially making a comparison between the linear and the non linear approach on the basis of the solutions obtained through numerical simulations.

In Appendices A and B are reported and described the codes and the procedures used for the mesh generation and the simulations.

Chapter 1

Mathematical Models

1.1 History

Since the early steps in tumor's growth studies, most of the models, were based on the assumption that cancer's cells divided at rates, varying among different type of tumors, that could be considered approximately constant on large temporal scales.

As described in [HAJS07] first studies on human cancers were conducted by Collins et al., observing the evolution of metastases in lungs through plain x-rays discovered that they grew at constant volume-doubling rate following an exponential law [CLT56]. This article had an incredible echo in the scientific community and in the years were proposed a lot of various relatively static models (see [AJ95], [AJ94b], [AJ91], [AJ94a] and [HTC01]). This kind of models are called "static" because they all basically lead to an exponential law, doesn't take in account any form of cell motility, making them suitable only to describe the evolution of a solid tumor, but totally inappropriate to describe the growth of cancers with a strong dynamic of invasion.

For a theoretical analysis of the relation between the cell kinetics and evolution of a cancerous mass, a pioneer is considered Steel [Ste77], with his works he started to look at the different time scales that characterize the cell and the gross kinetics. In this model were formulated several equations, each of them arithmetically simple involving a large number of parameters: T_p (the potential doubling time), T_d (the actual gross volume- doubling time), T_c (the cell cycle time), T_s (the time during which DNA is being synthesized within T_c), γ (a factor defining the timing of T_s within T_c and the age distribution of the cells), LI (the labeling index), φ (the cell loss factor, the proportion of newly formed cells that are lost), and GF (the growth fraction, the proportion of the total cells that are actually cycling). Alvord and Shaw, in [AJS91], developed further Steel's work adding to these unknowns the amount of tumor mass (mostly invisible) left behind after surgical resection and creating a nomogram that approximated the interrelations between most of Steel's unknowns. Exploiting simple arithmetic of Steel's equations, they were able to include survival time. Survival time could either be predicted or, more generally, be used retrospectively to estimate how much tumor that was all or mostly invisible to the neurosurgeon or neuroradiologist that had been left behind at the time of treatment. On the basis of these works Wilson could, in [Wil92],estimate the gradient of glioblastoma cells as exponential and calculate its concentration decreasing. After these pioneeristical studies on tumor's growth dynamics (especially glioma cells evolution), a solid base for a mathematical model began to take form. The creation of modern models was conducted with a step–

by–step procedure, in which every new model presented new features in a process of continuing transformation and modification. Every newly proposed model provided necessary improvements in order to adjust deficiencies of the previous ones. In the early 1990's Murray et al (see [Mur03])proposed a spatio–temporal model. It was conceived for the glioma, but most of its characteristics are suitable for most brain cancers. The basic model considers the evolution of the cancer's cells to be mainly governed by diffusion and proliferation. As in previous proposed models, the cells are assumed to duplicate following an exponential law. This hypothesis are reasonable, and reflect the biological processes involved at the time scale with which Murray was concerned, the time of disease's evolution. For this reason this type of model doesn't take in account necrosis, because its effects are significant only in highly proliferative, and slowly diffusive tumors (however the model could be easily modified to include it). This model is simplified by the assumption that brain tumors rarely metastasize outside the brain, making possible to describe the phenomenon by a balance law:

rate of change of cancer's cells density =
diffusion of cancer's cells in grey and white matter +
proliferation of cancer's cells.

The models presented in this work are essentially based on the equation proposed by Murray with some modifications.

1.2 Model Derivation

1.2.1 Basic Model

As seen in [Mur03] we consider $c(\mathbf{x}, t)$ as the number of cancer's cells at position $\mathbf{x}$ and time t. The model could be written, in dimensional form, as a balance equation:

$$\frac{\partial c}{\partial t} = \nabla \cdot \mathbf{J} + \rho c,$$

indicating with $\rho(time^{-1})$ the net rate of cells growth, accounting proliferation and death, and where $\mathbf{J}$ is the diffusive flux of cells. Under the assumptions of classical gradient–driven Fickian diffusion, $\mathbf{J}$ is taken proportional to the gradient of cell density:

$$\mathbf{J} = D\nabla c,$$

D $(length^2/time)$ is the diffusion coefficient of tumor's cells in brain tissues. At this stage the governing equation is:

$$\frac{\partial c}{\partial t} = D\nabla^2 c + \rho c. \tag{1.1}$$

The brain tissue is here considered homogeneous, so that diffusion and growth rates are taken constant among the domain. To deal with the complex structure of the brain the model has to be modified.

1.2.2 Spatial Heterogeneous Model

Brain tissues have different biomechanical properties and in particular in this context, the tumor cells diffusion is to be considered. From *in vivo* studies, regarding glioma cells diffusion conducted on rats (Chicoine and Sibergeld 1995, Chicoine and Sibergeld 1997, Kelley and Hunt 1994), it appeared that white matter serves as a channel of invasion between grey matters areas for cancer's cells. These results suggest to take a larger motility coefficient for cells in white matter than in grey matter.

On the basis of the models proposed by Swanson et al. (see [Swa99], [SAJM92], [SAJ02], [SAJM02a], [SAJM03] and [SAJM02b]), the model can be modified to allow diffusion coefficients to vary spatially among brain areas, so introducing a description of the complex geometry of the brain. The equation (1.1) still applies but the model now involves spatially varying diffusion $D(\mathbf{x})$,

$$\frac{\partial c}{\partial t} = \nabla \cdot (D(\mathbf{x})\nabla c) + \rho c, \tag{1.2}$$

where usually $D(\mathbf{x})$ has the following form:

$$D(\mathbf{x}) = \begin{cases} D_w & \text{for } \mathbf{x} \text{ in white matter} \\ D_g & \text{for } \mathbf{x} \text{ in grey matter} \end{cases} \tag{1.3}$$

with $D_w > D_g$, reflecting the different diffusive properties of the white and the grey matter.

The determination of the diffusion coefficients used in (1.3) has been the subject of many article and studies. One technique is described in [SBMAJ03] and consists in relating v, velocity of detectable tumor boundary, with ρ (cancer cells proliferation rate) and a random walk diffusion coefficient D obtained using Fisher's approximation (see [BKMAJ97] and [Mur03]):

$$D = \frac{v^2}{4\rho}. \tag{1.4}$$

This approximation comes from the observation that a population governed by growth and diffusion expands at a velocity of $2\sqrt{\rho D}$ for large time and is linear for a given ρ and D.

From examination of medical images, precisely CT scans, is derived the rate of advance of the tumor margin within the grey and the white matter, v_g and v_w, respectively. The analysis conducted in [TCW$^+$95] and [TCW$^+$96], studying the margin of detectable tumor movement, gave an average velocity $v = 0.008\ cm/day$. Using as usual in these cases a net proliferation $\rho = 0.012/day$, relation (1.4) suggests the diffusion coefficient $D = v^2/4\rho = 0.0013\ cm^2/day$. Being these data obtained from an invasion front in the proximity of the deep cerebral nuclei (predominantly grey matter), this value is associated to D_g. So, having $v_g = v = 0.008\ cm/day$, and considering that from the CT scans, the speed of advance of the tumor margin across the corpus callosum (white matter) is two to three times faster than in (predominately) grey matter; is possible to estimate that $v_w > 2.2v_g$ at $0.018\ cm/day$ and $D_w > 5D_g$ giving as diffusion coefficient $D(\mathbf{x})$:

$$D(\mathbf{x}) = \begin{cases} D_w = 0.0065\ cm^2/day & \text{for } \mathbf{x} \text{ in white matter} \\ D_g = 0.0013\ cm^2/day & \text{for } \mathbf{x} \text{ in grey matter} \end{cases} \tag{1.5}$$

To complete the phenomenon description, it is mandatory to add a boundary condition to the model. From the hypothesis assumed in Section 1.1, knowing that rarely brain tumors metastasize outside the brain, a null flux boundary condition is imposed. So, considering B as the brain domain on which (1.2) is to be solved, the model become:

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (D(\mathbf{x})\nabla c) + \rho c & \text{in } B \times (0,T) \\ \mathbf{n} \cdot D(\mathbf{x})\nabla c = 0 & \text{on } \partial B \times (0,T) \\ c_{|t=0} = c_0(\mathbf{x}) & \text{in } B \end{cases} \tag{1.6}$$

where ∂B is the boundary of the brain domain and $0 < T < +\infty$. The initial condition $c_0(\mathbf{x})$ describes the cancer cell distribution in the brain at time $t = 0$ and has the following structure:

$$c_0(\mathbf{x}) = \begin{cases} C_0 & \text{for } \mathbf{x} \text{ in cancer zone} \\ 0 & \text{otherwise} \end{cases}$$

where C_0, is a positive constant derived from the $MRIs$ analysis.

1.3 Linear Diffusion Model

1.3.1 Model Formulation

Starting from the model (1.20) presented in previous Section, is here implemented, with some modifications, a model suitable to describe the growth of a medulloblastoma near to the ventricles under the *corpus callosum*.

The first modification to be made is to add a factor that considers the area of the brain where a medullablastoma grows. Ventricles are filled with cerebrospinal fluid, so modelling the evolution of cancer cells in this zone makes mandatory to modify properly the diffusion term ((1.3)) dealing with the presence of the fluid. The cell motility could be described with a $D(\mathbf{x})$ of the following form:

$$D(\mathbf{x}) = \begin{cases} D_w & \text{for } \mathbf{x} \text{ in white matter} \\ D_g & \text{for } \mathbf{x} \text{ in grey matter} \\ D_{csf} & \text{for } \mathbf{x} \text{ in cerebrospinal fluid} \end{cases} \tag{1.7}$$

where the newly introduced coefficient is taken $D_{csf} \gg D_w > D_g$, this relation is derived from the observation of the dynamics of medulloblastoma growth (in the test case has been chosen a value of $D_{csf} = 0.1000\ cm^2/day$).

One problem analyzing this type of model is to determine the initial cells density of the cancer at t=0, $c_{|t=0} = c_0(\mathbf{x})$. To avoid this complication, the equation (1.2) could be divided by C_0 (knowing that $C_0 \geq 0$ as seen above), thus obtaining an equation for the non–dimensional quantity $0 \leq \tilde{c}(\mathbf{x}, t) \leq 1$, where $\tilde{c}(\mathbf{x}, t)$ now indicates the concentration of cancer cell in $\mathbf{x}$ at time t:

$$\begin{cases} \dfrac{\partial \tilde{c}}{\partial t} = \nabla \cdot (D(\mathbf{x}) \nabla \tilde{c}) + \rho \tilde{c} & \text{in } B \times (0, T) \\ \mathbf{n} \cdot D(\mathbf{x}) \nabla \tilde{c} = 0 & \text{on } \partial B \times (0, T) \\ \tilde{c}_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.8}$$

where $k(\mathbf{x})$ is

$$k(\mathbf{x}) = \begin{cases} 1 & \text{for } \mathbf{x} \text{ in cancer zone} \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

Working with $\tilde{c}(\mathbf{x}, t)$ force to modify properly also the reaction term of the equation, in fact using a linear reaction term doesn't ensure that $\tilde{c}(\mathbf{x}, t) \leq 1 \; \forall \mathbf{x}$ and $\forall t$. This could be solved using a different reaction term, however this modification is treated later in this work.
Models like (1.8) belong to the class of linear parabolic problems, studying this kind of system of partial differential equations makes necessary to establish whether the problem has a solution, and if this solution exists analyze its uniqueness.

1.3.2 Model Analysis

Considering (1.8), following analysis steps proposed in [Sal], first of all is necessary to find a weak formulation. The primary step is to multiply for a sufficiently regular function $\phi = \phi(\mathbf{x})$ the first equation of the system, integrating on the brain domain B, indicating to ease the notation $\tilde{c}$ as c from now on:

$$\int_B \frac{\partial c(\mathbf{x}, t)}{\partial t} \phi(\mathbf{x}) d\mathbf{x} - \int_B \nabla \cdot (D(\mathbf{x}) \nabla c(\mathbf{x}, t)) \phi(\mathbf{x}) d\mathbf{x} - \int_B \rho c(\mathbf{x}, t) \phi(\mathbf{x}) d\mathbf{x} = 0.$$

Applying integration by parts to the second term we get:

$$\int_B \frac{\partial c(\mathbf{x}, t)}{\partial t} \phi(\mathbf{x}) d\mathbf{x} - \int_{\partial B} \mathbf{n} \cdot D(\mathbf{x}) \nabla c(\mathbf{x}, t) \phi(\mathbf{x}) d\mathbf{x} + \\ + \int_B D(\mathbf{x}) \nabla c(\mathbf{x}, t) \cdot \nabla \phi(\mathbf{x}) d\mathbf{x} - \int_B \rho c(\mathbf{x}, t) \phi(\mathbf{x}) d\mathbf{x} = 0.$$

Enforcing the boundary condition of null flux of the system (1.8), the integral on ∂B is equal to zero, thus obtaining the following weak formulation for the problem:

$$\int_B \frac{\partial c(\mathbf{x},t)}{\partial t}\phi(\mathbf{x})d\mathbf{x}+\int_B (D(\mathbf{x})\nabla c(\mathbf{x},t))\cdot\nabla\phi(\mathbf{x})d\mathbf{x}-\int_B \rho c(\mathbf{x},t)\phi(\mathbf{x})d\mathbf{x}=0. \quad (1.10)$$

At this point it is useful to consider $c = c(\mathbf{x}, t)$ as a function that for any value $t \in [0, T]$ gives a function of $\mathbf{x}$, in an appropriate Hilbert space V, $c : [0, T] \to V$. So using this convention is possible to use $c(t)$ instead of $c(\mathbf{x}, t)$, resulting from (1.10)

$$\int_B \dot{c}(t)\phi d\mathbf{x}+\int_B (D(\mathbf{x})\nabla c(t))\cdot\nabla\phi d\mathbf{x}-\int_B \rho c(t)\phi d\mathbf{x}=0. \quad (1.11)$$

It's now mandatory to find a proper functional environment, a suitable Hilbertian tern (V, H, V^*). Since an appropriate choice is to take $c(t) \in H^1(B)$, it's quite natural from the theory to pose $H = L^2(B)$ and $V^* = H^{-1}(B)$, so having $H^{-1}(B)$ as natural space for $\dot{c}(t)$. It's known from the functional analysis of Sobolev spaces (see [Sal]), that since, from the spaces choice, are $c \in L^2([0, T]; V)$ and $\dot{c} \in L^2([0, T]; V^*)$ it follows that

$$c \in C([0, T]; H)$$

making possible to identify $H = L^2$ a suitable space for the initial cell density condition k.

Using the commonly adopted notation and interpreting in the duality sense the first integral of (1.11), and introducing the following bilinear form:

$$a(w, \phi) = \int_B (D(\mathbf{x})\nabla w) \cdot \nabla \phi d\mathbf{x} - \int_B \rho w \phi d\mathbf{x}. \tag{1.12}$$

is now possible to write the weak formulation of the problem (1.8):

Find a function $c \in L^2([0,T]; V)$ *such that* $\dot{c} \in L^2([0,T]; V^*)$ *and*:

1. *for any* $\phi \in V$,
$$\langle \dot{c}(t), \phi \rangle_* + a(c, \phi) = 0 \tag{1.13}$$
a.e in $[0, T]$;

2. $c(0) = k$.

Once a weak formulation is given, it is possible to prove that the problem (1.8) has one and only one existing solution.

The bilinear form (1.12) is at least weakly coercive, in fact it is always possible to choose a value $\lambda \geq 0$ and $\alpha > 0$ such that for any $v \in V$:

$$a(v, v) + \lambda \|v\|_H^2 \geq \alpha \|v\|_V^2$$

where $\|\cdot\|_H^2$ and $\|\cdot\|_V^2$, are the norm obtained using the internal product, in H and V respectively. The above property gives the possibility to implement, following the analysis conducted in [Sal] for evolution problems, the well known Faedo-Galerkin method. This method grants that the problem (1.13) is well posed under the assumptions made, and ensures the existence and the uniqueness of the solution for the system (1.8) (in this work the complete implementation is omitted, see always [Sal] for details).

1.3.3 Model Properties

As seen in the previous Sections, the model (1.8) is a quite simple instrument to describe the complex behavior of the brain cancer cells. Describe the growth of the tumor using a diffusion–reaction equation for the cells concentration, gives the possibility to analyze and synthesize molecular and cellular properties of tissues from a macroscopic point of view. All the information on the biological processes involved in the cell proliferation and diffusion trough the tissues is contained in the diffusion coefficient $D(\mathbf{x})$ and in the reaction one, ρ. In [SBMAJ03] there's a brief analysis of the role played by these coefficients, especially regarding the ratio between them. From the assumptions made, the cell concentration depends on the $\rho/D(\mathbf{x})$. An interesting consequence of this fact is that, varying $D(\mathbf{x})$ and ρ, while keeping their ratio fixed, the geometry of the tumor growth and invasion remains the same, only the time scale on which the growth and invasion occurs is changed. Specifically, two different tumors with $\rho/D(\mathbf{x})$ fixed could appear exactly the same at a single observation time step but could reach their ultimately fatal size in very different lengths of time. This illustrates the clinical complications of using only one MRI observation of the lesion and proceeding with treatment without really knowing the expected pattern of growth. Anyway is possible to obtain cells concentration profiles for a specific tumor on the basis of a model only by knowing the value of the above coefficients.

Is mandatory to observe the fact that studying this phenomenon with a system of partial differential equation like (1.8), we are forced to modify the obtained

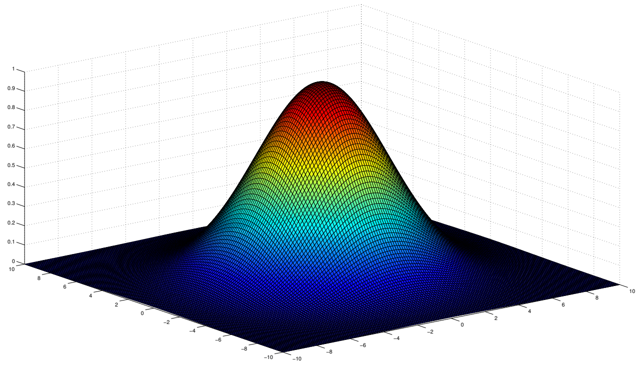


Figure 1.1: Solution of the linear equation in a square domain.

solution in order to give it a physical sense. The model presented in the above Section has a fundamental solution (see [Sal]), e.g solving the system in a square domain, gives the result observed in Figure [1.1]. The remarkable fact to notice is that this kind of solutions has a problem to deal with. The cells diffusion in linear models, like (1.8), doesn't reproduce the propagation front of cancer cells in tissues. In fact looking at this equations, a positive concentration of cancer cells could be found also far from the main mass of the tumor. This could be possible looking at the tumor in the advanced phases of its growth, but in modeling these phenomenon is necessary to consider a model that better highlight the tumor borders. To deal with this problem is possible, as done also by Swanson et al., to put threshold beyond which the concentration of cancer cells is retained negligible, see Figure [1.2] for an example in the square case.

Another fact that spots out studying this model, is that cell proliferation process modeled in this way, doesn't take in account the fact that the tissues in which the cancer grows deteriorate and modify their diffusive properties during the prolifer-

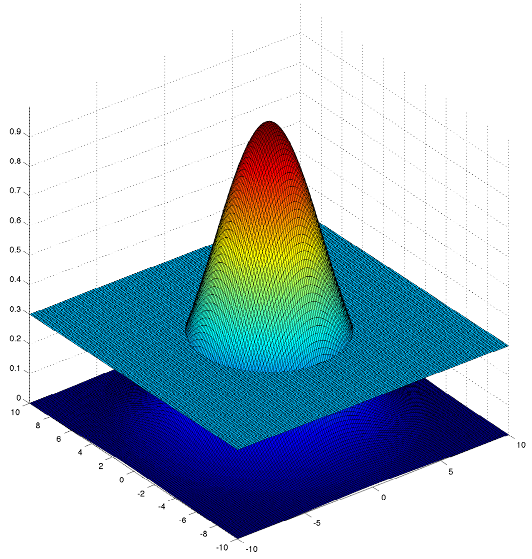


Figure 1.2: Solution with a threshold to avoid non physical behavior.

ation. It's obvious that a partially invaded tissue has different diffusive properties respect to a sane or a total invaded one. Now the aim will be to modify properly the model (1.8) to consider this factors and model in a more physical way the phenomenon.

1.4 Non Linear Diffusion Model

As seen in the previous Section, modeling tumor growth using linear equation to describe cancer's cells behavior, could be a smart and effective technique but has a major problem to deal with, treating the diffusion in a linear way leads to obtain non physical solution and is based on strict assumptions that couldn't be justified.

Is possible to modify in a quite simple way the existing equation to consider these factors. The idea followed in this work is based on a "conversion" of the linear equation to a porous medium equation where a non linearity is introduced in the diffusive term.

1.4.1 Model Overview

Let's consider the linear model (1.8), and modify the diffusion term introducing a power $m > 0$ of the cell concentration as follows, for details see [Sal] for the derivation of these models:

$$\frac{\partial c}{\partial t} = \nabla \cdot (mc^{m-1} D(\mathbf{x}) \nabla c) + \rho c.$$

Where all the parameters are the same of the linear case, and the diffusion coefficient $D(\mathbf{x})$ has the same form of (1.7).

As seen in [Vaz12], the power of the cancer cell concentration plays a fundamental role in the implementation of the model. The exponent m varies mainly in the range $m > 1$, obtaining the linear case for $m = 1$. It is even possible to extend it to the fast diffusion range (FDE) $m < 1$ under the condition that $m > mc = max[(N-2)/N, 0]$, where N is the dimension in which the problem is studied. The complete form of the non linear model for the evolution of the cancer cell concentration is:

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (mc^{m-1} D(\mathbf{x}) \nabla c) + \rho c & \text{in } B \times (0,T) \\ \mathbf{n} \cdot D(\mathbf{x}) \nabla c = 0 & \text{on } \partial B \times (0,T) \\ c_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.14}$$

The first things to notice is that the non linearity introduced gives a degenerate diffusion term: when the cell concentration decreases to value near to zero the diffusion vanishes.

1.4.2 Model Analysis

Here the analysis is carried on following the work done in [Vaz07], in order to obtain the well posedness for the non linear model. In the following we start considering a model without reaction term, further the well posedness is extended to diffusion–reaction cases.

Pure Diffusion Case

First of all, let's introduce a suitable concept of weak solution, for the problem without reaction term, defining $B_T = B \times (0,T)$ the following system is obtained:

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (mc^{m-1} D(\mathbf{x}) \nabla c) & \text{in } B_T \\ \mathbf{n} \cdot (mc^{m-1} D(\mathbf{x}) \nabla c) = 0 & \text{on } \partial B \times (0,T) \\ c_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.15}$$

Posing for ease the notation $\nabla \cdot (\nabla \Phi(c)) = \nabla \cdot (mc^{m-1} D(\mathbf{x}) \nabla c)$. Notice that the function $\Phi(c)$ satisfies the properties required in [Vaz07]:

- $\Phi : \mathbb{R}_+ \longmapsto \mathbb{R}_+$ is continuous and strictly increasing in c with $\Phi(0_+) = 0$;
- $\Phi(\pm\infty) = \pm\infty$;
- $\Phi(c)$ is smooth with $\Phi'(c) > 0$ for $c > 0$.

Definition 1. A weak solution of equation (1.14) in B_T is a locally integrable function, $c \in L^1_{loc}(B_T)$, such that

- $z = \Phi(c) \in L^1([0,T]; W^{1,1}(B))$;
- c satisfies the identity

  $$\int_B \int_0^T (\nabla \Phi(c) \cdot \nabla w - c w_t) d\mathbf{x} dt = \int_B c_0(\mathbf{x}) w(\mathbf{x}, 0) d\mathbf{x}$$

  $\forall w \in C^1(\bar{B}_T)$ which vanishes for $t = T$.

It's possible to prove the following.

Theorem 1. *Under the additional assumptions that* $\Phi(c) \in L^2([0,T]; H^1(B))$ *and*

$c \in L^2(B_T)$. The problem (1.14) has at most one weak solution.

It is now possible to define the primitive of Φ

$$\Psi(s) = \int_0^s \Phi(t)dt.$$

Concerning the initial data, such a setting leads us to assume that c_0 is a measurable function such that $\Psi(c_0(\mathbf{x})) \in L^1(B)$. Let's call this space $L_\Psi(B)$. When c_0 belongs to $L_\Psi(B)$ is possible to develop a theory for energy weak solutions.

Theorem 2.

- *Under the stated conditions on Φ, assume that $c_0 \in L_\Psi(B)$. Then the problem (1.14) has a weak energy solution defined in an infinite time interval, $T = \infty$. So $c \in L^\infty((0,T); L_\Psi(B))$ and $\Phi(c) \in L^2([0,T]; H_0^1(B))$, and the energy inequality*

  $$\int_B \int_0^T |\nabla \Phi(c)|^2 d\mathbf{x}dt + \int_B \Psi(c(\mathbf{x},T))d\mathbf{x} \leq \int_B \Psi(c_0(\mathbf{x}))\mathbf{x}$$

  holds. Moreover, $c \in C([0,+\infty); L^1(B))$.

- *For every two solutions c and $\bar{c}$ with initial data c_0, $\bar{c_0}$ resp., for every $t > \tau \geq 0$ holds*

  $$\|(c(t) - \bar{c}(t))_+\|_1 \leq \|(c(\tau) - \bar{c}(\tau))_+\|_1. \tag{1.16}$$

where $\| \cdot \|_1$ indicates the L^1–norm. Following the steps made in [Vaz07] it is also possible to extend the whole theory to general initial data that belongs to L^1.

Theorem 3. *Let Φ be a monotone function as above. Then, for any $c_0 \in L^1(B)$*

there exists a unique $c \in C([0,\infty); L^1)$ *that solves problem* (1.15) *in the sense of limit solutions. The weak energy solutions of the theorem 2 are limit solutions. The map* $c_0 \longmapsto c$ *is an ordered contraction from* $L^1(B)$ *into* $C([0,\infty); L^1(B))$ *in the sense that* (1.16) *holds for limit solutions.*

Let's write the limit solution $c(t)$ obtained for data $c_0 \in L^1(B)$ as $S_t c_0$. Then:

Corollary. *The map* S_t *define a continuous semigroup of contractions in* $L^1(B)$*, and* S_t *preserves* $L^1(B)_+$*. The semigroup is uniformly bounded.*

In this way the problem (1.15) is proved to be well posed and positive preserving in L^1.

Diffusion–Reaction Case

To extend the results obtained in the pure diffusion case to the problem (1.14), it is possible to introduce a proper change of variables.
We consider the following substitution:

$$\begin{cases} \bar{c} = e^{-\rho t} c; \\ \tau = \dfrac{1 + e^{(m-1)\rho t}}{(m-1)\rho}. \end{cases} \tag{1.17}$$

Substituting (1.17) in (1.14) and remembering that ρ is a fixed constant, we obtain:

$$\frac{\partial \bar{c}}{\partial \tau} = \nabla \cdot (m \bar{c}^{m-1} D(\mathbf{x}) \nabla \bar{c}). \tag{1.18}$$

Applying the same analysis made in the pure diffusion case on (1.18), it is possible to obtain the well posedness also for the diffusion–reaction case.

1.4.3 Model Properties

As seen in Section 1.3, the model is essentially governed by the values of the diffusion and proliferation coefficients, particularly by the ratio between them. Also the value of the coefficient m is important to describe the growth process. The more evident difference consists in the non linearity introduced in the diffusion term. In this way the solution of the system has a more physical behavior. The solutions of these kind of equations highlights the sharp propagating front and better describes the growth of a solid mass with defined boundaries. Another interesting feature is that introducing in the diffusion a term that depends on the cancer cells concentration means physically to describe also the degeneration of the invaded tissues, modifying their properties when varying the value of cell concentration.

In the following analysis the model will be studied posing $m = 2$, resulting

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (2cD(\mathbf{x})\nabla c) + \rho c & \text{in } B \times (0,T) \\ \mathbf{n} \cdot D(\mathbf{x})\nabla c = 0 & \text{on } \partial B \times (0,T) \\ c_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.19}$$

The value of the coefficient is chosen on the basis the speed of the tumor front in the MRIs used for the numerical simulation. Studying the variation of the mean diameter over the MRI slices on time, it is possible to obtain a proper value for

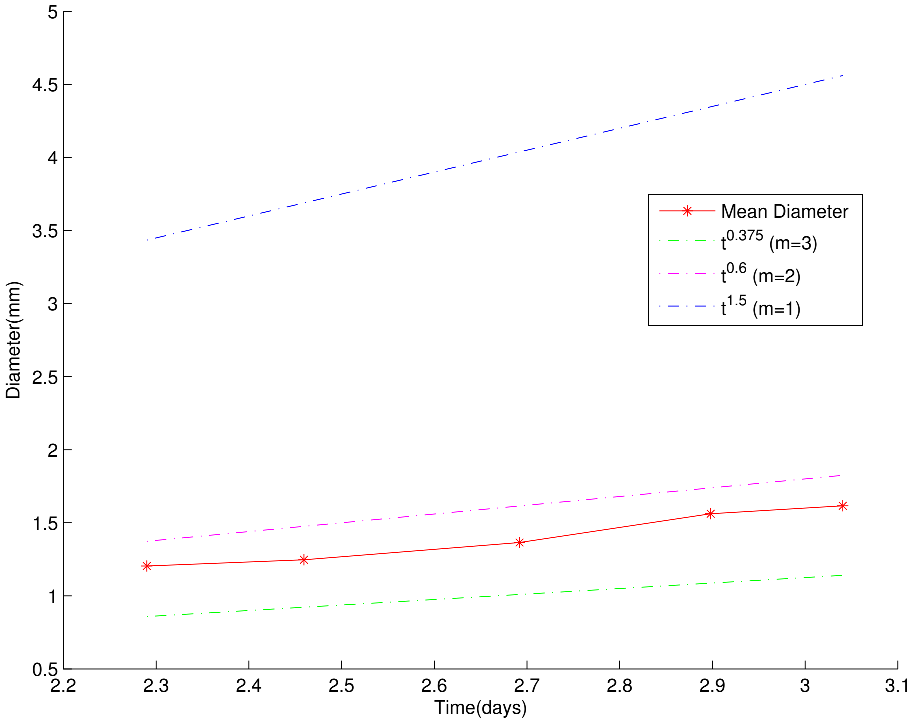


Figure 1.3: Mean diameter over MRI layers on time.

the exponent of the cell concentration in the diffusion term, in order to describe the proliferation of tumor cells following their behavior basing on the variation of the cancer mass during its evolution process. From [Vaz07] it is known that solutions of the non linear equation has particular scaling properties, and from them is possible to estimate that the radius of the growing mass follows the rule

$$r(t) \sim t^{\alpha}$$

where

$$\alpha = \frac{N}{N(m-1)+2}$$

considering that the problem is studied in three dimension ($N = 3$), the choice of m gives a value for the exponent $\alpha = 0.6$, that is a good approximation according to the diameter evolution studies, see Figure [1.3].

1.5 Reaction Term

As discussed in Section 1.3, the use of a linear reaction term, it is not a proper choice when the cancer's growth is described by studying the evolution of its cells concentration, because it doesn't ensure to obtain a bounded solution.Indeed, to model the phenomenon in a proper way, is necessary to consider a function $c(\mathbf{x}, t) \leq 1$. To ensure the preceding inequality a further modification of the diffusion reaction models has been suggested, as done and suggested in [JMD^{+}05]. This modification consist in modeling the reaction term as a function $f(c)$ depending on the number of cancer cells that represent the temporal evolution pattern

of the growth. In this way the models obtained, introducing such a reaction term, remember the equations proposed in [GM77] for biological populations.
The function $f(c)$ takes different forms:

$$f(c) = \rho c$$

$$f(c) = \rho c \log \frac{c_{max}}{c}$$

$$f(c) = \rho c(c_{max} - c)$$

In the present work, the function chosen to describe the growth pattern, is a logistic law of the following form $f_{log}(c) = \rho c(1 - c)$.
The linear model become:

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (D(\mathbf{x})\nabla c) + f_{log}(c) & \text{in } B \times (0, T) \\ \mathbf{n} \cdot D(\mathbf{x})\nabla c = 0 & \text{on } \partial B \times (0, T) \\ c_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.20}$$

while the non linear is now

$$\begin{cases} \dfrac{\partial c}{\partial t} = \nabla \cdot (2cD(\mathbf{x})\nabla c) + f_{log}(c) & \text{in } B \times (0, T) \\ \mathbf{n} \cdot D(\mathbf{x})\nabla c = 0 & \text{on } \partial B \times (0, T) \\ c_{|t=0} = k(\mathbf{x}) & \text{in } B \end{cases} \tag{1.21}$$

The property of well posedness for both the systems presented remains, and it can be proven using iterative or fixed point techniques.

Chapter 2

Numerical Models

The models presented in the previous Chapter are governed by linear and non linear parabolic equations. For the simulations was used the finite element method with Galerkin discretization.

2.1 Linear Model

2.1.1 Galerkin Formulation

Consider the following bilinear forms taking c,z and ϕ in $V = H^1(B)$:

$$m(c, \phi) = \int_B c\phi d\mathbf{x};$$

$$a(c, \phi) = \int_B D(\mathbf{x})\nabla c\nabla\phi d\mathbf{x};$$

and the trilinear form

$$r(z,c,\phi) = \int_B c(1-z)\phi d\mathbf{x}.$$

The problem (1.20) could be now written as

$$m(\dot{c},\phi) + a(c,\phi) + r(c,c,\phi) = 0, \qquad \forall \phi \in V$$

remembering as seen in Section 1.3 that $c = c(t)$.
Let's consider the spaces

$$\begin{aligned} X_h^r &= \left\{ v \in \mathbb{C}^0,\, v_{|_K} \in \mathbb{P}_r \ \forall K \in \mathcal{T}_h \right\}, \\ V_h &= \left[H^1(B) \cap X_h^r \right]; \end{aligned}$$

$\mathcal{T}_h$ is a tetrahedralization of the domain B, and indicating with $\{\varphi_j\}$ a base for the space V_h is possible to introduce an approximation for c:

$$c_h = \sum_i^{N_h} c_i \varphi_i.$$

The problem can be now written using the approximation c_h and the proper spaces introduced before

$$m(\dot{c_h},\phi_h) + a(c_h,\phi_h) + r(c_h,c_h,\phi_h) = 0, \qquad \forall \phi_h \in V_h \tag{2.1}$$

with $\dot{c}_h = \sum_i^{N_h} \dot{c}_i \varphi_i$.

2.1.2 Algebraic Formulation

Defining $\mathbf{c} = (c_1, c_2, ..., c_{N_h})^T$ and the following matrices

$$\mathbf{M} = [m_{ij}] = m(\varphi_j, \varphi_i),$$
$$\mathbf{A} = [a_{ij}] = a(\varphi_j, \varphi_i),$$
$$\mathbf{R}(z) = [r_{ij}(z)] = r(z, \varphi_j, \varphi_i);$$

is possible to obtain a matricial formulation of the equation (2.1)

$$\mathbf{M}\dot{\mathbf{c}} + (\mathbf{A} + \mathbf{R}(z))\mathbf{c} = 0.$$

For the time discretization a *Backward Euler* scheme is used. Indicating with $\Delta t = t^{k+1} - t^k$ the time discretization step and and choosing at every time step a semi–implicit treatment of the reaction term by posing $z = c^k$, the formulation become:

$$\mathbf{M}\frac{\mathbf{c^{k+1}}}{\Delta t} + (\mathbf{A} + \mathbf{R}(\mathbf{c^k})\mathbf{c^{k+1}} = \mathbf{M}\frac{\mathbf{c^k}}{\Delta t},$$

Notice that at every time step the problem solved is linear. For further details on discretization,stability and convergence properties see [Qua08].

2.2 Non Linear Model

The most challenging quest in using this model is to treat the non linearity in the diffusion term in a proper way. To deal partially with it is used a fixed point technique. Let's see how the formulation of the problem change.

2.2.1 Galerkin Formulation

First of all consider the trilinear form $\tilde{a}(\cdot,\cdot,\cdot)$, always taking c,w and ϕ in $V = H^1(B)$:

$$\tilde{a}(w, c, \phi) = \int_B 2wD(\mathbf{x})\nabla c\nabla\phi d\mathbf{x}.$$

Using same spaces and notation of Section 2.1, the problem (1.21) become

$$m(\dot{c_h}, \phi_h) + \tilde{a}(c_h, c_h, \phi_h) + r(c_h, c_h, \phi_h) = 0, \qquad \forall\phi_h \in V_h.$$

2.2.2 Algebraic Formulation

In order to obtain a full discrete formulation, consider the matrix

$$\tilde{\mathbf{A}}(w) = [\tilde{a}_{ij}(w)] = \tilde{a}(w, \varphi_j, \varphi_i)$$

As done for the linear model an implicit time discretization is used, treating the reaction term in a semi–implicit way, leading to the matricial formulation

$$\mathbf{M}\frac{\mathbf{c^{k+1}}}{\Delta t} + (\tilde{\mathbf{A}}(w) + \mathbf{R}(\mathbf{c^k}))\mathbf{c^{k+1}} = \mathbf{M}\frac{\mathbf{c^k}}{\Delta t}. \tag{2.2}$$

At every time step the algebraic problem (2.2) has to be solved. As said before the non linearity is treated using a fixed point technique posing at the first step of the fixed point iteration $w = \mathbf{c}^k$. A convergence criterion based on the distance in L^2 between the solution at two consecutive steps is used:

$$e^p = \|c^{p+1} - c^p\|_{L^2(B)} \leq tol$$

where *tol* is a convergence threshold and p represent the fixed point iteration step. In this way the problem solved at any fixed point iteration is linear and it can be solved as in the previous case using iterative methods. To avoid the problem of the oscillations provided by the steep gradients near the zones where the cells concentration goes to zero, the solution is modified at every time step posing to zero the concentration below a threshold.

Chapter 3

Numerical Implementation

3.1 Mesh Generation

Simulate phenomena in biological process forces to describe complex domains. It is then essential to create well refined meshes close to the reality of the problem without exceed in computational costs. In the present work the modeling of the brain domain has been performed using VTK (*Visualization Tool Kit*, *http://www.vtk.org*) and *vmtk* (*Vascular Modelling Tool Kit*, *http://www.vmtk.org*), through a *Python* interface, and working on a series of MRIs of a patient provided by the INT (*Istituto Nazionale Tumori*, *"Banca Dati www.tumori.net"*).

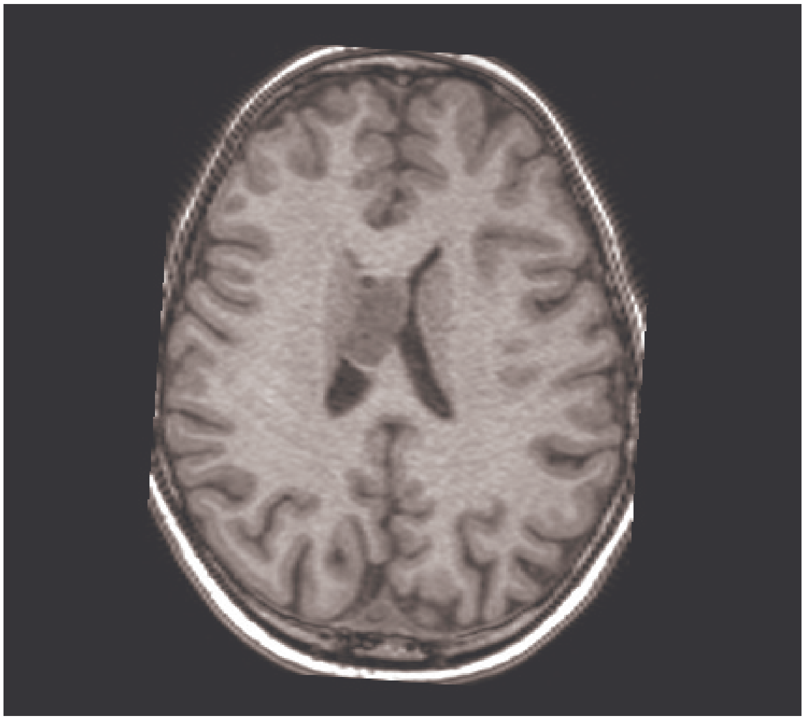


Figure 3.1: Example of MRI, trasversal slice.

3.1.1 Medical Images Segmentation

The output of a MR (*Magnetic Resonance*) is a grey scale image, e.g. Figure [3.1]. The grey intensity is correlated to the tissue's diffusion properties. To obtain information about tissues and their features, it is necessary to segment the brain areas putting labels on different regions. In the simulation carried out applying the models presented in Chapter 1, the segmentation process has the aim to identify five different areas of the image:

- *Grey Matter*:
- *White Matter*;

- *Cerebro Spinal Fluid*;
- *Tumor*;
- *Background.*

Indeed the interest is to have a map of the brain, from which is possible to detect the location of the brain's boundary, to estimate the local values of diffusion $D(\mathbf{x})$ (defined in (1.7)) and where the tumor is located.

The procedure is performed using fuzzy logic segmentation and classification. To provide an automatic identification of the area of interest a fuzzy set classification is used, basing on the T1 grey level intensity of the brain tissues. For every area of the brain that is needed for the domain description, it is defined a membership function using a Gaussian kernel centred on a proper diffusion value. The values for the center of these kernels and their standard deviation are optimized by expectation maximization on a subset of the tissue of the brain considered. The results of this fuzzy segmentation are five maps that report the probability of each pixel to belong to the five tissue types. The values of these maps range from 0 to 100, where 100 represents the maximum probability to belong to a particular discrete fuzzy set label. By comparing the values of the different maps, and putting a threshold on the probability of belonging to a certain tissue, is possible to create a single label map. The obtained map describes the location of the tissues in the brain domain, and contains all the information needed for the numerical implementation of the model.

3.1.2 Mesh Creation

The creation of the computational mesh is divided in two phases:

- Grid creation and refinement
- Label assignment

after these operations the computational domain is ready to be used for the numerical simulations. Let's see the whole process in more detailed way

Grid Creation and Refinement

Starting from the MRI, where the skull has already been removed, a surface using *vmtk* is extracted, excluding the background of the image. After some processes of cleaning and filtering a smooth and closed surface is obtained. The surface is now remeshed with a triangulation. For the remeshing phase is very important to work with regular objects without holes or sharp edges. Once this process is completed, a tetrahedral mesh is generated using *Tetgen* (*http://tetgen.berlios.de/*). The resulting unstructured grid is properly refined near the area where the cancer grows: in this way the computational cost are kept the same increasing the precision in the area of interest. In Figures [3.2], [3.3], [3.4] and [3.5] the final mesh is represented.

Label Assignment

After the mesh refinement, it is necessary to superpose the segmented map to the grid, in this way the information on geometry and the regions position of

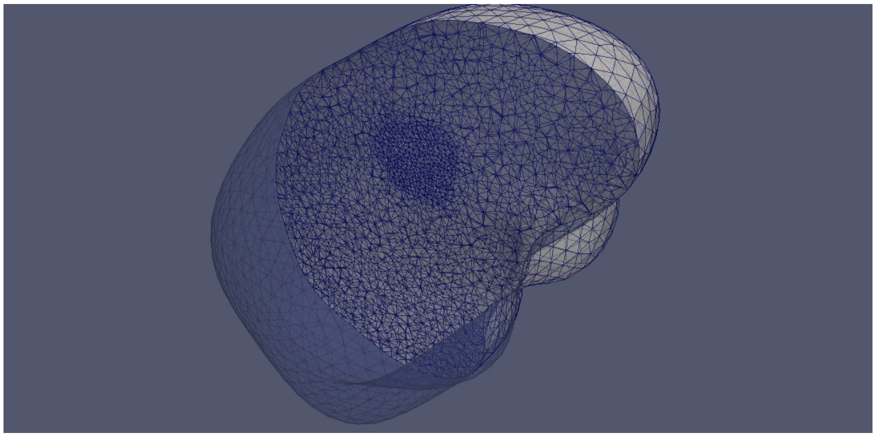


Figure 3.2: A view of the mesh.

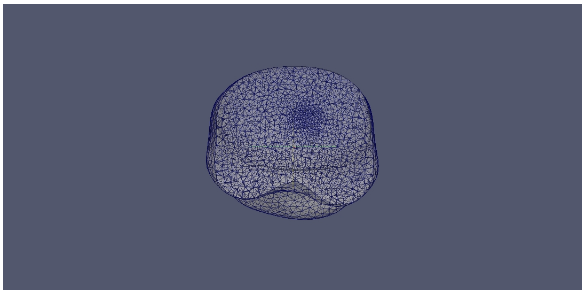


Figure 3.3: Coronal view of the mesh.

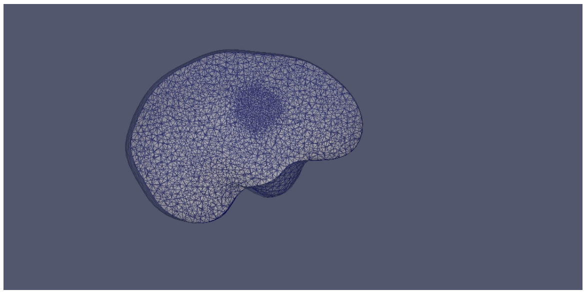


Figure 3.4: Sagittal view of the mesh.

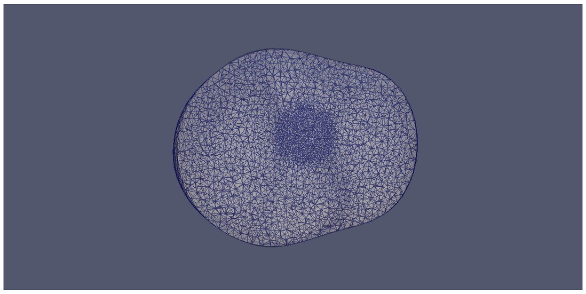


Figure 3.5: Trasversal view of the mesh.

the MRI are passed to the brain mesh. This operation is performed using VTK and the C++ library *MhdOrientation* (for details on the implementation and the development of the library see Appendix B). The grid and the label map are superimposed assigning to every cell of the mesh the value of the map in the corresponding point. Every label of the map is related to a diffusion value, that at the end of this phase is assigned to the unstructured grid. During this operation it is also created the initial condition putting the value one in the cells corresponding to the cancer invaded areas, and zero in the others (for an example of the mesh compared to the map see Figure [3.6]). Once these steps are done, the mesh is ready for the simulation and contains all the properties necessary to implement the diffusion coefficients based models. All the codes used for the phase of mesh generation are reported in Appendix A.1.

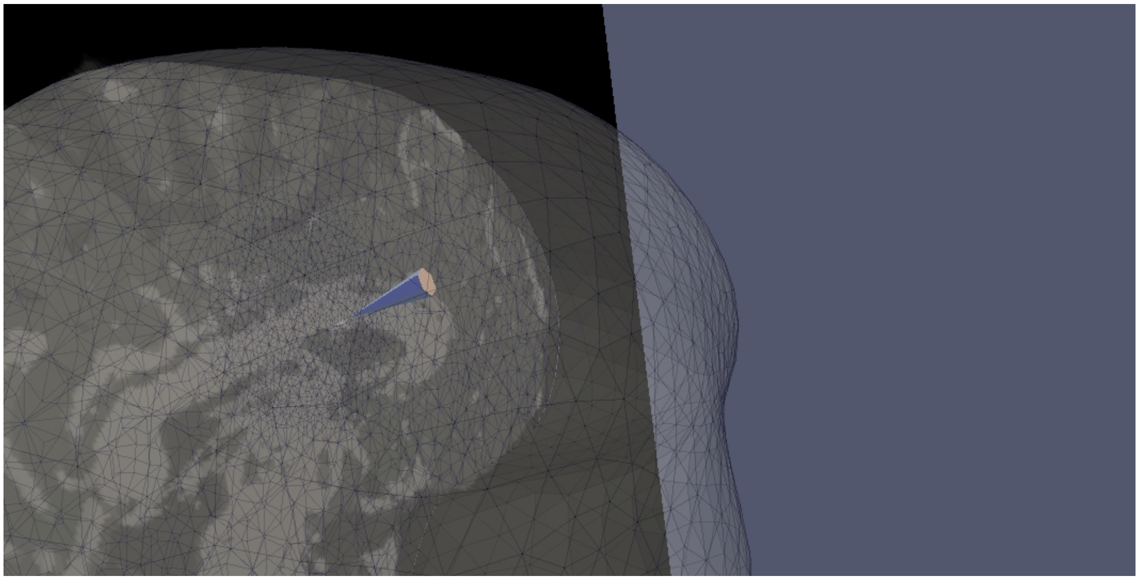


Figure 3.6: A detail of the mesh superimposed on the segmented map in the cancer area.

3.2 Software and Implementation Details

3.2.1 *FreeFem++* Implementation

All the simulations performed in this work were executed using the software *FreeFem++* (*http://www.freefem.org/ff++/*). Its interface with the *VTK* libraries, is essential for the brain mesh loading. However not all the *VTK* features are implemented under *FreeFem++*; this is the case, in particular, for the cell data reading that is fundamental to obtain the diffusion information and the initial condition assigned to the the grid in Section 3.1. So it is necessary to pass this data to the code in a different way. In the phase of label assignment, discussed in the previous Section, two files are created: one containing all the diffusion values and another containing initial cancer's cells concentration. These files are loaded in execution phase as piece–wise constant functions defined on any element of the mesh. At every time step the function containing the diffusion coefficients is updated considering the cancer cell concentration in the examined tissue, in fact the diffusive properties of the brain regions are compromised when the tumor heavily invades a zone. To model this behavior the update of the diffusion function is done putting a threshold on the cell concentration, when it goes over a critical value the diffusion is changed. All the outputs are written in *VTK* format.

3.2.2 Data Visualization

The output data of the simulations are all examined using *VTK* and *Paraview.* To follow the growth of the tumor, the unstructured grid containing the value of cells concentration is clipped with slices where a contour plot is displayed. The segmented map of the brain at different phases of the cancer evolution is compared with those contour plots in both linear and non linear case.

Another comparison is based on the fraction of volume occupied by the tumor at different stages, the data are represented using *Gnuplot* through its interface in *Matlab*(*http://www.mathworks.com/matlabcentral/*). All the codes used for the simulations are included in Appendix A.2.

Chapter 4

Results

The results of the simulations of tumor growth using the two models, presented in Chapter 1 and Chapter 2, are reported in this Chapter. The scope of these simulations is to compare the results of the two models, and verify if the evolution predicted by the tests matches with the experimental data.

The data, provided by INT, consists in MRIs of a specific patient affected by a medulloblastoma located near the ventricles, under the *corpus callosum*. The images considered for the evaluation of the models are taken along a period of 16 months. The results are discussed focusing on two different aspects. In Section 4.1 the models are compared with the data by a superimposition of the simulation's results over a map, obtained from the segmentation of the medical images, at two different phases of the growth. In Section 4.2, the aim is to evaluate how the volume of the evolving cancer mass is consistent with the clinical observation.

All the simulations are performed on a tetrahedral mesh, generated following the steps shown in 3.1, with 188028 elemnts and 30393 vertices. Since for the tests was chosen a discretization using $\mathbb{P}_2$ finite elements, the number of degrees of freedom is equal to 250512.

4.1 Tumor Growth

In this Section is conducted an analysis working over the segmented maps. The VTK output, obtained from the code presented in Appendix A.2, is studied making a contour plot, on a threshold imposed on the non–dimensional cancer cell density equal to $1e-4$ (beyond this threshold the non linear model results remain unchanged), over three different slices(transversal, coronal and sagittal). The slices are chosen selecting planes that cut the cancerous spheroid in its central part to describe the entire dimension of the mass. As said in Section 3.2 all the images included to show the results are displayed using *Paraview*. The images considered for this study correspond to two different stages of the tumor growth, they are divided in two sets, for each sets are represented the linear and the non linear diffusion model compared to the experimental data. The first set of images refers to the evolution at 195 days, while the second makes reference to the segmented map related to the $MRIs$ of the patient at 492 days.

As seen, for example in Figures [4.1] and [4.7], the linear diffusion model overestimates the tumor mass size, as reported also by Swanson works([HAJS07], [JMD$^+$05] or [SBMAJ03]). The non linear diffusion's modification, reduces this exaggerated cells motility. This is evident looking at Figures [4.2] and [4.8]. In the non linear diffusion case the cells movement decreases in areas already invaded by the tumor. This follows from the introduction of cells density in the diffusion

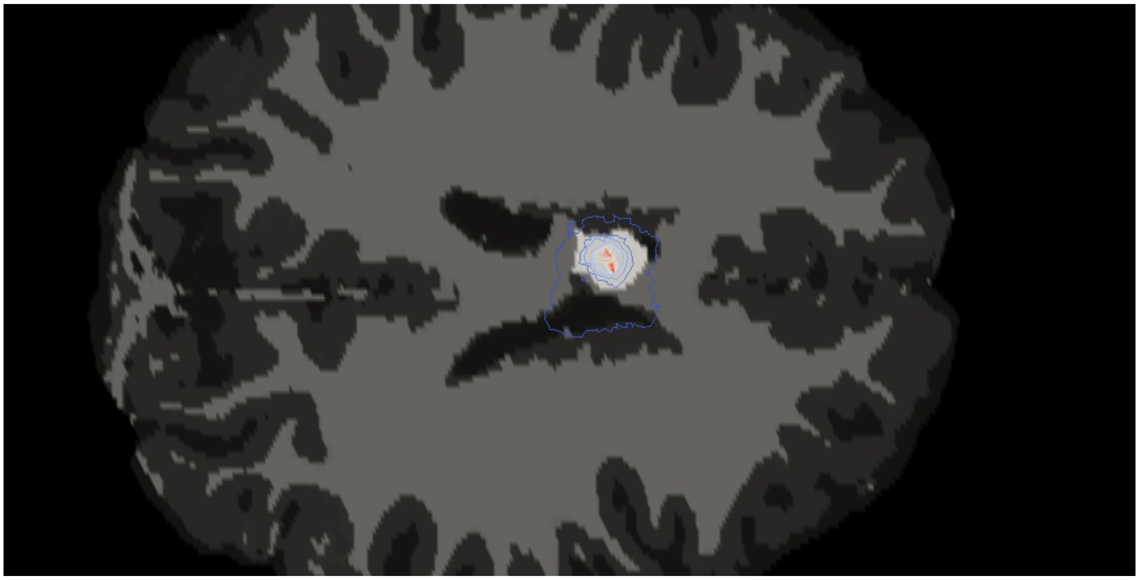


Figure 4.1: Transversal view of the contour plot at $t = 195$ days linear diffusion case.

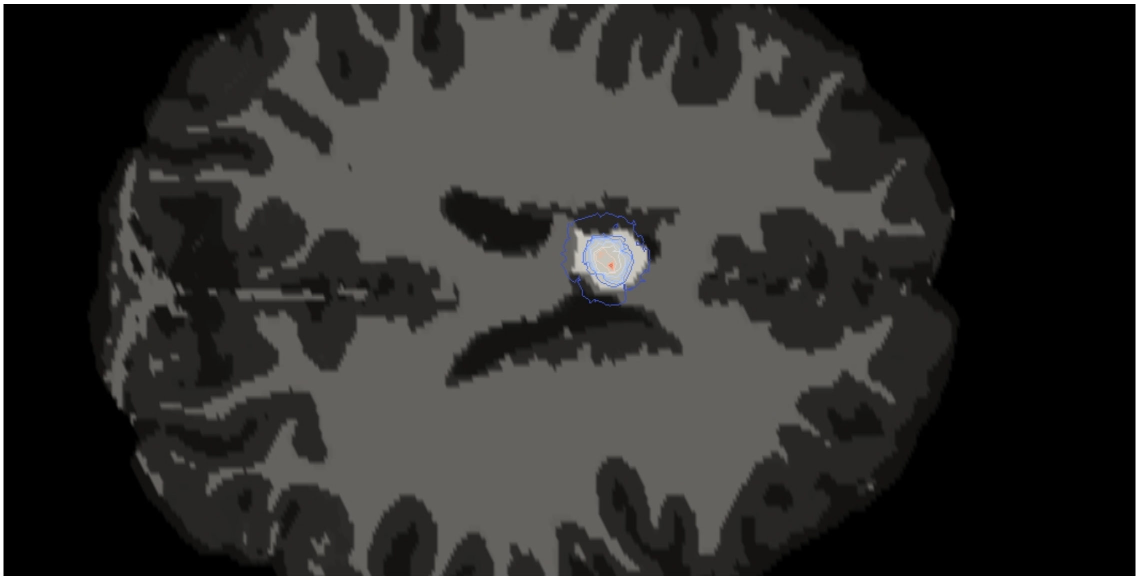


Figure 4.2: Transversal view of the contour plot at $t = 195$ days non linear diffusion case.

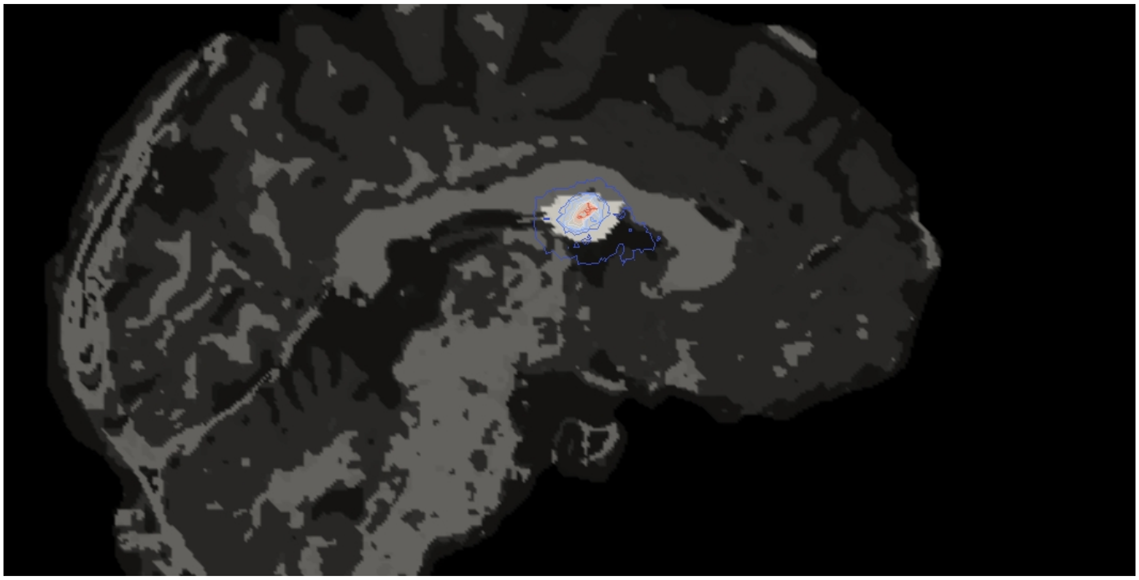


Figure 4.3: Sagittal view of the contour plot at $t = 195$ days linear diffusion case.

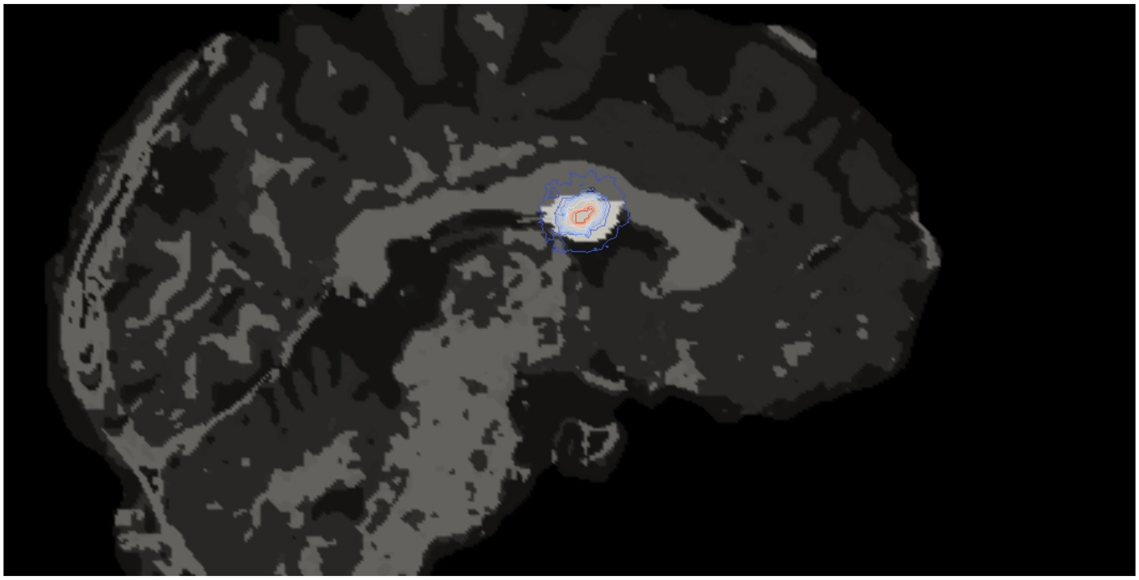


Figure 4.4: Sagittal view of the contour plot at $t = 195$ days non linear diffusion case.

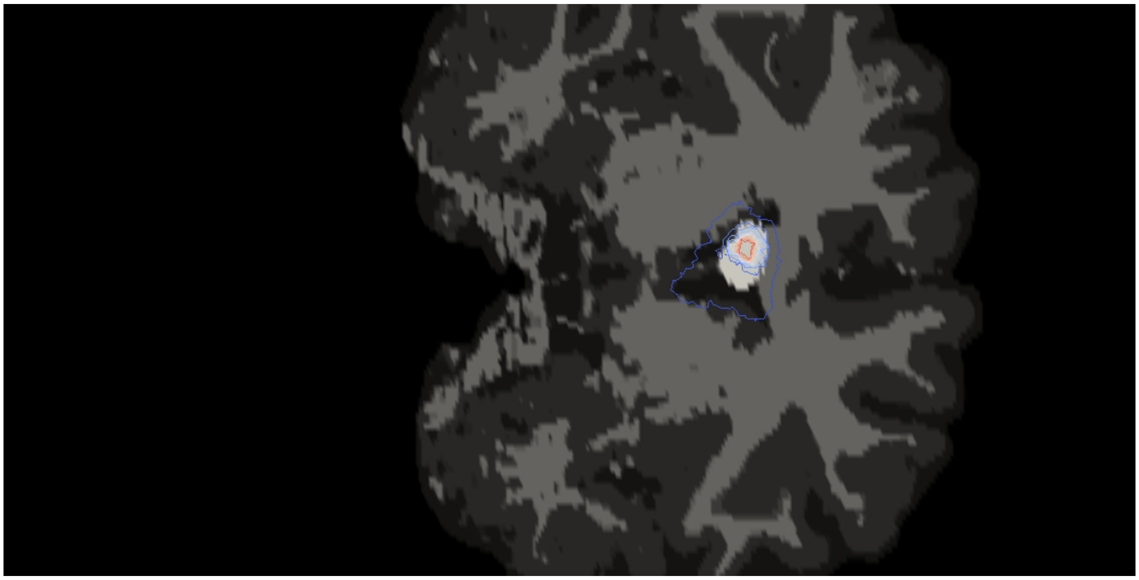


Figure 4.5: Coronal view of the contour plot at $t = 195$ days linear diffusion case.

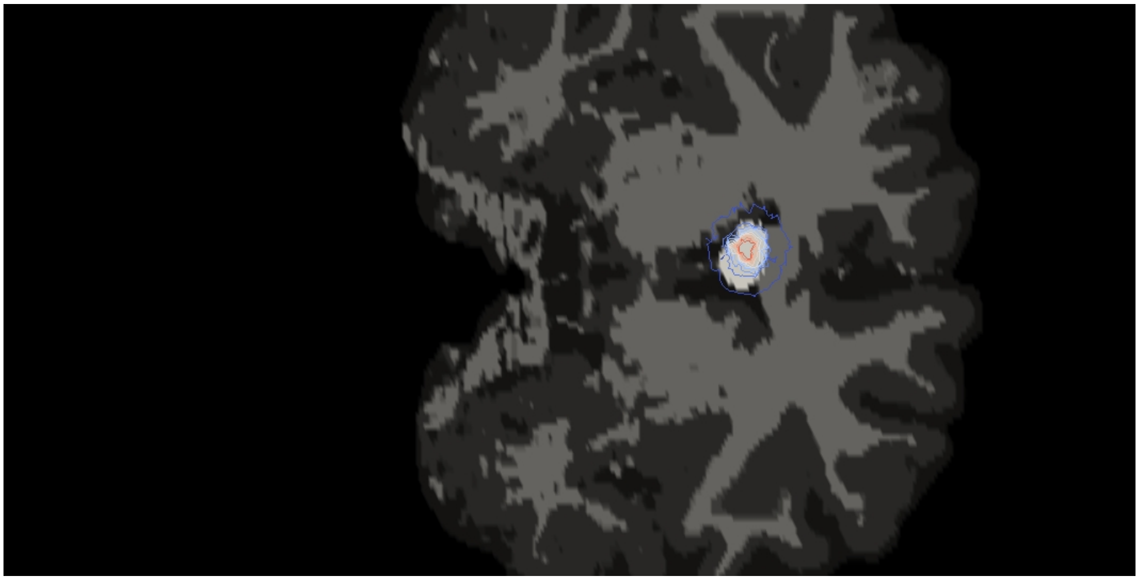


Figure 4.6: Coronal view of the contour plot at $t = 195$ days non linear diffusion case.

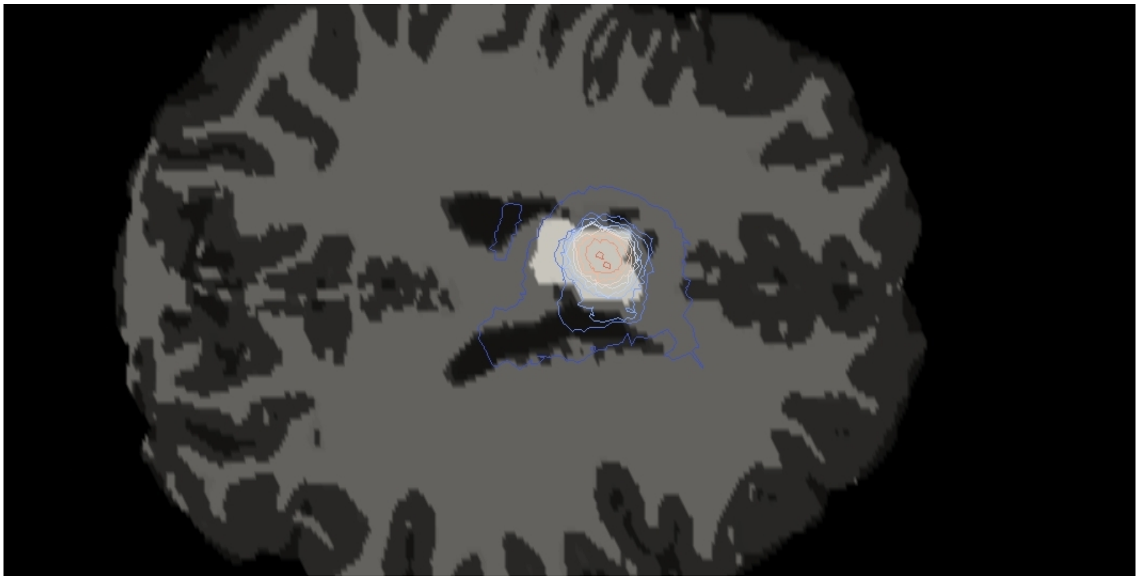


Figure 4.7: Transversal view of the contour plot at $t = 492$ days linear diffusion case.

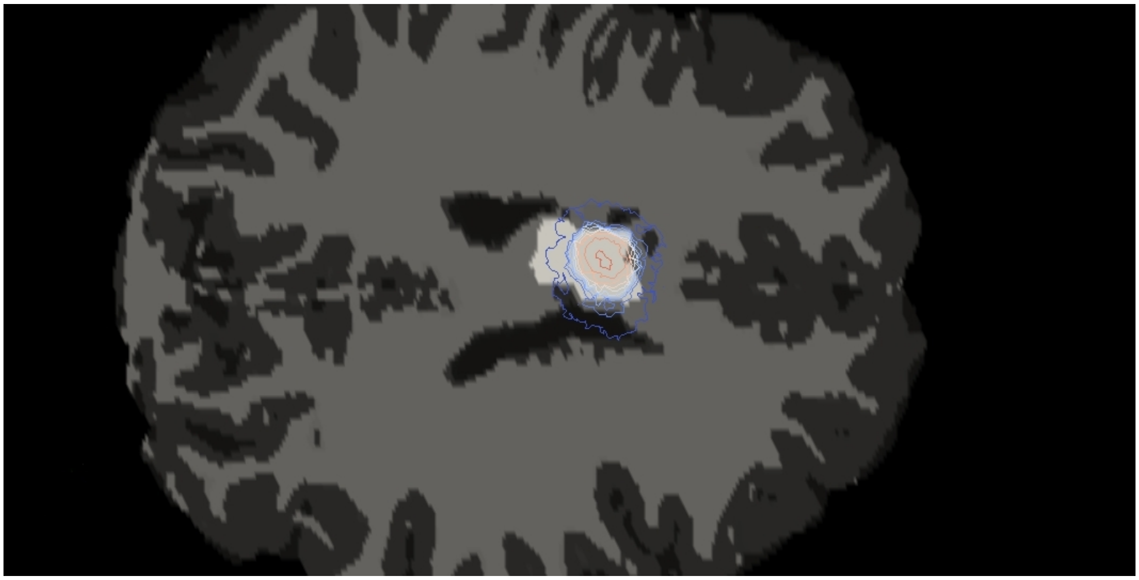


Figure 4.8: Transversal view of the contour plot at $t = 492$ days non linear diffusion case.

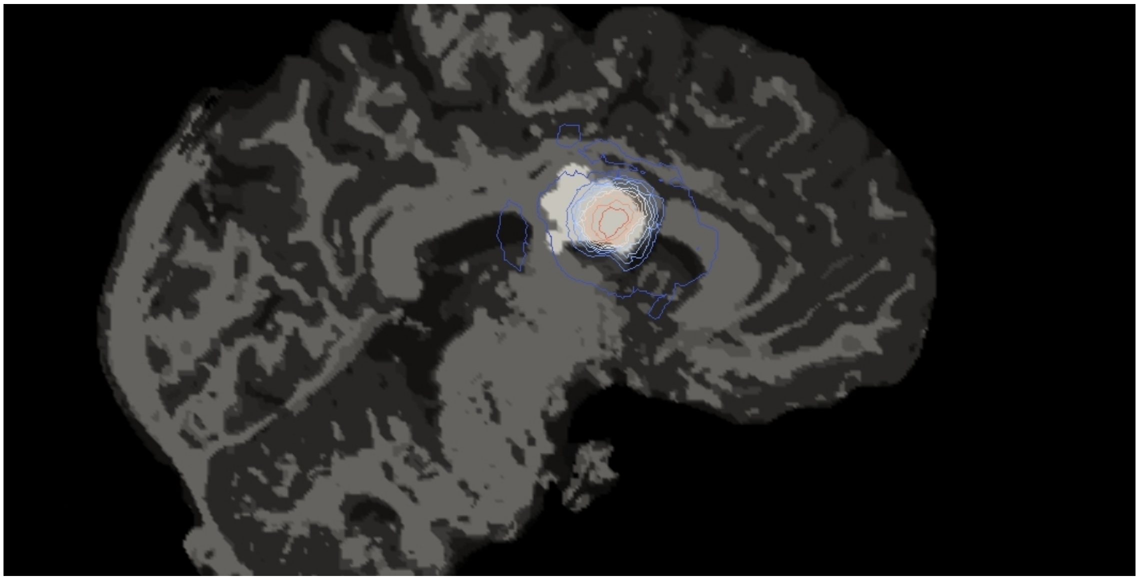


Figure 4.9: Sagittal view of the contour plot at $t = 492$ days linear diffusion case.

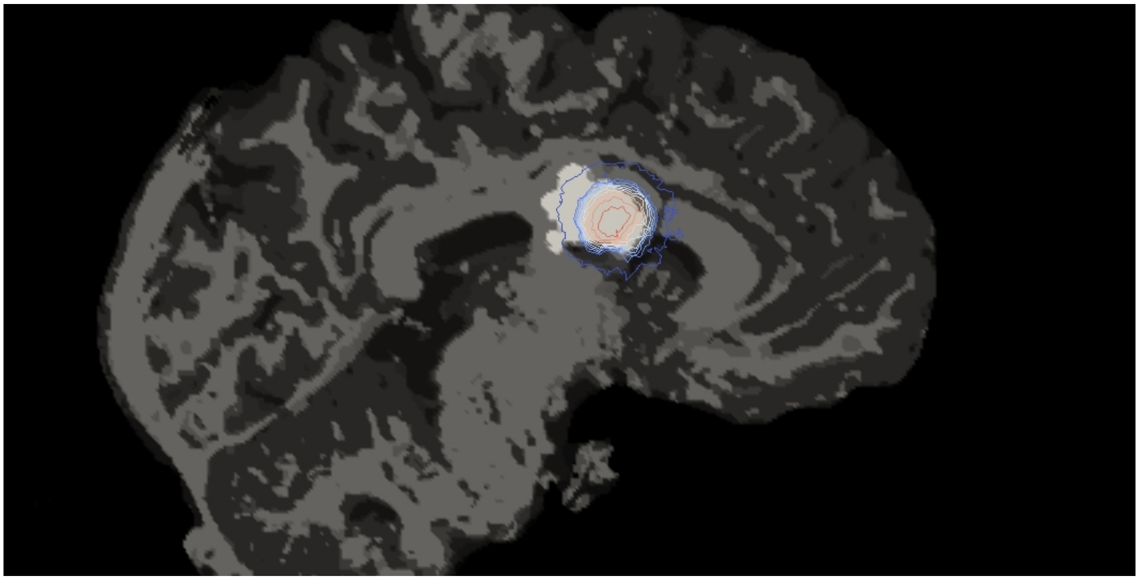


Figure 4.10: Sagittal view of the contour plot at $t = 492$ days non linear diffusion case.

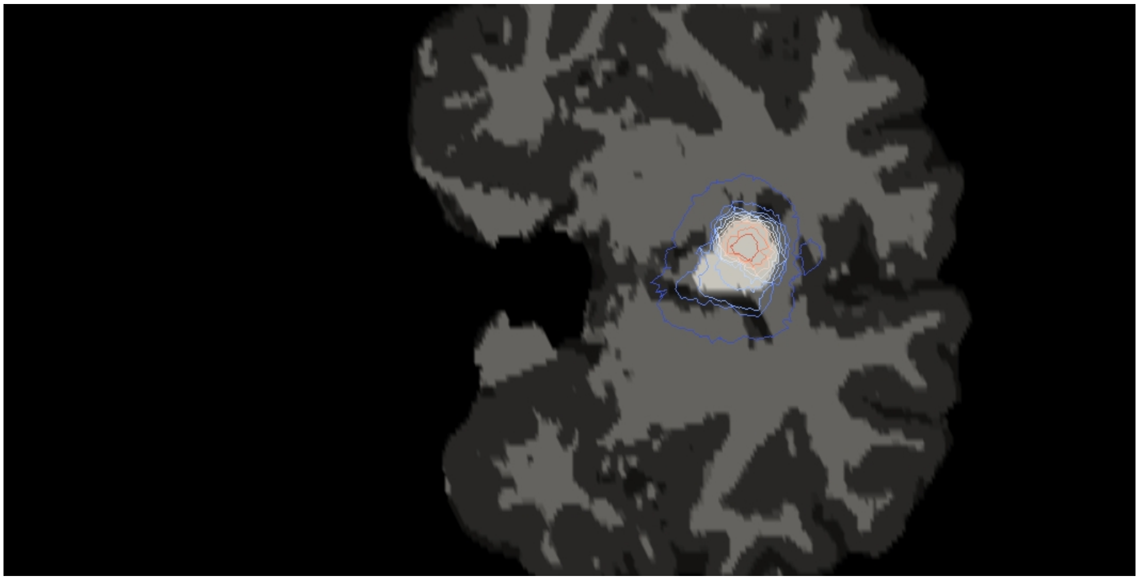


Figure 4.11: Coronal view of the contour plot at $t = 492$ days linear diffusion case.

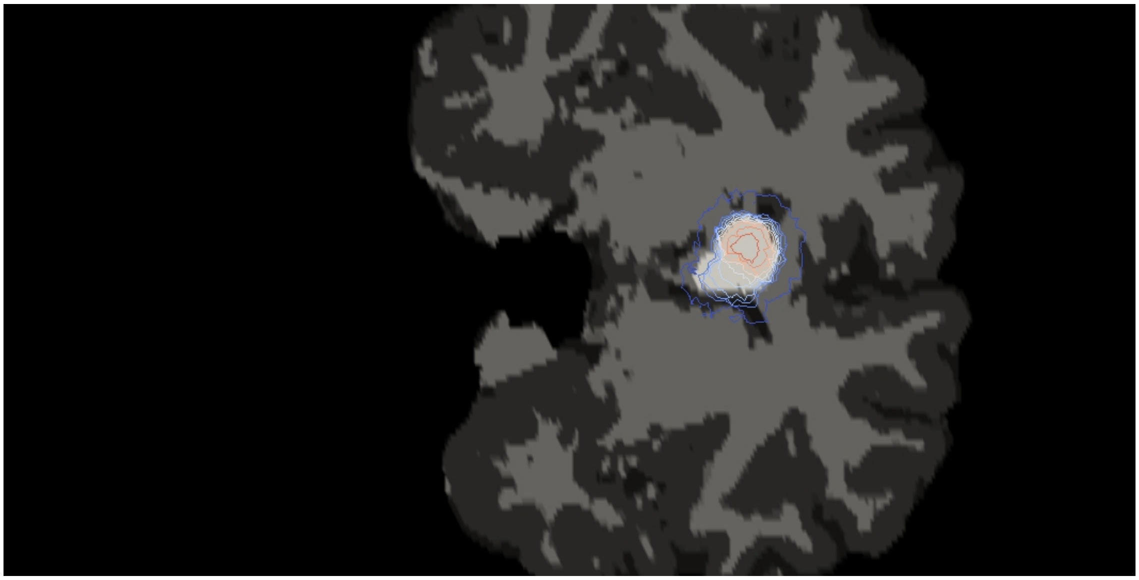


Figure 4.12: Coronal view of the contour plot at $t = 492$ days non linear diffusion case.

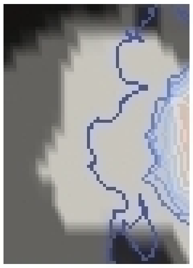


Figure 4.13: Detail of image [4.8].

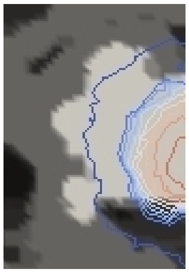


Figure 4.14: Detail of image [4.10].

term.

Considering all the Figures, that shows the results of the non linear model, is evident, particularly looking at the images at $t = 492$ days, that there's a discrepancy in the growth represented by the model and the data. The zooms in Figures [4.13] and [4.14] highlight that the model reproduce the correct flux of tumor cells properly in the ventricles area towards the posterior part of the brain. The failure in this area is probably due to the fact that the segmentation of this zone of the domain, as described in Section 3.1, is too rough and doesn't describe properly the structures involved in the process of cancer's growth.

T	Linear Model	Non Linear Model	MRIs Segmentation Data
195	41.1188	16.9968	8.1858
492	173.5311	50.8594	13.9298

Table 4.1: Comparison of volumes at different time of growth, with fixed threshold at $1e-4$*, the volume is expressed in* cm^3*.*

4.2 Volume Evaluation

The study of the growing volumes is carried out analyzing the output of the simulations using $MATLAB$ and *Paraview* combined with VTK. In Table 4.1 are reported the value of the tumor's volume at both different stage of growth. As seen in the results presented in Section 4.1, models overestimate the dimension of the tumor mass. The larger motility of the linear diffusion term model gives, looking at the volume, a big error on the estimation. The non linear diffusion term, as said in the previous section, decreases the over diffusive effect giving results more near to the data obtained from the segmentation of the MRI. To make a further comparison between the two models is made also another kind of analysis. Considering at every time step of the simulation, the following data, evaluating the fraction of volume occupied by the cancer mass in the whole brain:

$$\frac{\int_B \chi_c d\mathbf{x}}{\int_B 1 d\mathbf{x}} \tag{4.1}$$

where χ_c is:

$$\chi_c = \begin{cases} 1 & \text{for } c \geq 1e-4 \\ 0 & \text{for } c < 1e-4 \end{cases} \tag{4.2}$$

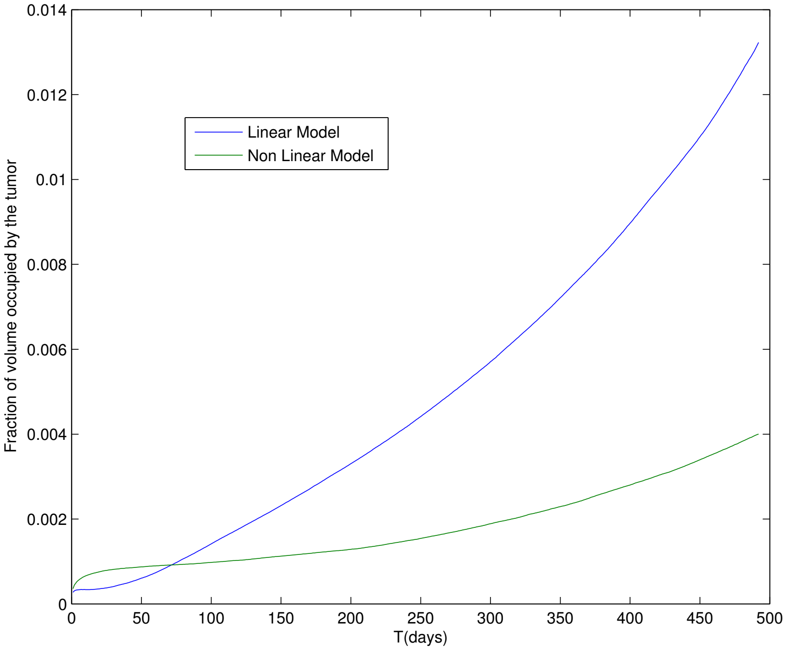


Figure 4.15: Comparison of the two models examining fraction of volume occupied by the cancer.

In Figure 4.15 is possible to evaluate the growth during the simulation of the tumor's volume, where is evident how the linear model overestimate the cells motility, while the usage of a non linear diffusion term helps to reduce considerably this effect.

Chapter 5

Conclusions

In this work were studied two different diffusion–reaction model in order to evaluate their ability to predict the growth of a brain tumor. Their properties were studied comparing the results, obtained through numerical simulations, and various label maps, created from a segmentation process on MRIs of a test patient. The data shown in Chapter 4, suggest that both models give interesting results in the description of brain tumor's evolution. As shown by the analysis carried on in Section 4.1 the models describe well the evolution of the tumor, while is the non linear diffusion model presents better results showing a closer relevancy to the experimental data. The application of the models presented in this work to the test case considered could be considered encouraging and suggests their intense use and study.

Both models are characterized by an overestimation of the cancer expansion, as seen in Section 4.2, but with a further work of calibration of the parameters we are confident that is possible to improve the accuracy of the models.

The discrepancy between simulations and data images could be also reduced by refining the segmentation process. Describing in a more detailed way the brain structure in the area of the growth should give a sensible improvement in the evolution modeled. Both models could be also modified introducing a further anisotropy in the diffusion term, considering a tensor that describes the fibers orientation in the brain's areas considered during the evolution of the cells mass. This will lead to consider also the structural properties of the brain, modeling the fact that cells motility in zone with strongly directional fibers tends to follow the way in which fibers are oriented. The code used for the simulations, see Appendix A.2, is already modified by the introduction of such a tensor in the diffusion term, setting the values of the tensor in the proper way will model the above mentioned conditions.

However is evident that a model like the linear one, though suitable to give an idea of the tumor growth, has a lot of limitations. Considering a diffusion term that doesn't depend from the cell concentrations lead to non physical solutions and doesn't describe the effect that invasion of the cancer has on the brain tissues. Introducing in the model an exponent of the cells concentration in the diffusion term accounts for the deterioration and consequentially improve the results keeping more compact the cancer's cells mass. This surely improves the detection of the tumor propagating front. Indeed the description of the growing tumor borders turns out to be crucial to understand their evolution and their behavior, particularly from a medical point of view. The price to be paid to use a non linear model, to describe better the physical behavior of the tumor, are the higher computational costs, due to the use of a fixed point technique. Needless to say that the use of a more suitable numerical scheme to model the non linear diffusion is mandatory, especially concerning the front detection and the treatment of the tumor boundaries.

The behavior of the two models suggests the possibility to combine them to describe different stages of the cancer evolution. The non linear diffusion is suitable to describe the early phases of brain invasion, it tends to illustrate an evolution of a compact mass keeping a low cell motility, indeed when the tumor starts to grow is difficult to find cells far from the main mass. Instead in advanced phases of the tumor's growing process is possible to find cells far from the main mass, this fact makes the linear diffusion model's properties more suitable to describe this trend.

Starting from these models could be interesting also to consider further modifications, particularly to solve the problem of a clear definition of tumor's borders. One possibility could be to implement another modified version of the linear diffusion model, using a model similar to the Cahn-Hillard equations, that substantially consists in an introduction of a bilaplacian term that helps to define more precisely the contours of the growing cancers.

Another way to analyze and use the models presented in this work is to find an expression for the velocity of the tumor boundary, that makes possible to implement zero level set methods. This methods are particularly suitable to deal with growth of objects with defined boundaries, indeed they describe only the evolution of the front using an hyperbolic equation with a known speed of propagation. Obviously for this kind of models is necessary to implement and study proper numerical methods different from the ones used in this work.

Another issue of interest that could be researched in further works is to obtain, starting from a certain stage of the tumor growth, information on cancer's early stages of development, basing on the suggested models. This kind of procedures are known as inverse problems, and also in this case a whole theory, and also *ad hoc* numerical methods must be implemented. These studies could help to understand where and when the disease started, observing the phenomenon when it was not yet detectable via magnetic resonance, making possible to observe the tumor in its early phases of evolution.

Appendices

Appendix A

Code Used

A.1 *Python* Code

Meshing Phase

```python
#Meshing.py
from vmtk import pypes
from vmtk import vmtkscripts
from vmtk import vtkvmtk
import vtk
import sys
reader=vtk.vtkXMLPolyDataReader()
reader.SetFileName(sys.argv[1])
reader.Update()
surf=reader.GetOutput()
stm=vmtkscripts.vmtkSurfaceToMesh()
stm.Surface = reader.GetOutput()
stm.Execute()
#sizing mesh elements
sizingFunctionArray=vtk.vtkDoubleArray()
sizingFunctionArray.SetName("SizingFunction")
sizingFunctionArray.SetNumberOfTuples(stm.Mesh.GetNumberOfPoints())
stm.Mesh.GetPointData().AddArray(sizingFunctionArray)
for i in range(stm.Mesh.GetNumberOfPoints()):
  sizingFunctionArray.SetTuple1(i,4)
#call tetgen
tetgen=vmtkscripts.vmtkTetGen()
tetgen.Mesh= stm.Mesh
tetgen.GenerateCaps = 0
tetgen.UseSizingFunction = 1
tetgen.Refine=0
tetgen.SizingFunctionArrayName = "SizingFunction"
tetgen.Order = 1
tetgen.Quality = 1
tetgen.PLC = 1
tetgen.NoBoundarySplit = 1
tetgen.RemoveSliver = 0
tetgen.OutputSurfaceElements = 0
tetgen.OutputVolumeElements = 1
tetgen.Execute()
#mesh
writer=vtk.vtkUnstructuredGridWriter()
writer.SetFileName("Mesh_uni.vtk")
writer.SetInput(tetgen.Mesh)
writer.Update()
```

Mesh Refinement

```
#Refinement.py
from vmtk import vtkvmtk
from vmtk import pypes
from vmtk import vmtkscripts
import vtk
from math import sqrt,pow
reader = vtk.vtkUnstructuredGridReader()
reader.SetFileName("Mesh_uni.vtk")
reader.Update()
mesh=reader.GetOutput()
pb=[0.0,0.0,0.0]
```

```
pt=vtk.vtkPoints()
sizingFunctionArray=vtk.vtkDoubleArray()
sizingFunctionArray.SetName("newSizingFunction")
sizingFunctionArray.SetNumberOfTuples(mesh.GetNumberOfPoints())
mesh.GetPointData().AddArray(sizingFunctionArray)
for i in range(mesh.GetNumberOfPoints()):
  pb=mesh.GetPoint(i)
  dist=sqrt(pow((pb[0]-35),2) + pow((pb[1]-303),2)+ pow((pb[2]+40),2))
  if dist <= 20:
    sizingFunctionArray.SetTuple1(i,1)
  else:
    sizingFunctionArray.SetTuple1(i,2.5)
tetgen = vmtkscripts.vmtkTetGen()
tetgen.Mesh = mesh
tetgen.GenerateCaps = 0
tetgen.UseSizingFunction = 1
tetgen.Refine=1
tetgen.SizingFunctionArrayName = "newSizingFunction"
tetgen.Order = 1
tetgen.Quality = 1
tetgen.PLC = 0
tetgen.NoBoundarySplit = 1
tetgen.RemoveSliver = 1
tetgen.VarVolume = 1
tetgen.OutputSurfaceElements = 0
tetgen.OutputVolumeElements = 1
tetgen.Execute()
writer = vtk.vtkUnstructuredGridWriter()
writer.SetFileName("Mesh_adapt.vtk")
writer.SetInput(tetgen.Mesh)
writer.Update()
```

Label Assignment

```
#Label.py
from vmtk import vtkvmtk
from vmtk import pypes
from vmtk import vmtkscripts
```

```
import vtk
from math import sqrt,pow
def barycenter(points,pbar):
  nop=points.GetNumberOfPoints()
  for ii in range(0,3):
    summ=0.0
    for jj in range(nop):
      summ= summ + (points.GetPoint(jj))[ii]
    pbar[ii]= summ/nop
  return
reader = vtk.vtkUnstructuredGridReader()
reader.SetFileName("Mesh_tran.vtk")
reader.Update()
mesh=reader.GetOutput()
pb=[0.0,0.0,0.0]
pt=vtk.vtkPoints()
ptlst=vtk.vtkIdList()
ireader = vtk.vtkMetaImageReader()
ireader.SetFileName("Ras_res_3.mhd")
ireader.Update()
img=ireader.GetOutput()
ploc=vtk.vtkPointLocator()
ploc.SetDataSet(img)
ploc.AutomaticOn()
ploc.BuildLocator()
iml=img.GetPointData().GetScalars()
ncells=mesh.GetNumberOfCells()
cell_label=vtk.vtkDoubleArray()
cell_label.SetNumberOfTuples(ncells)
cell_label.SetName("cell_labels")
mesh.GetCellData().AddArray(cell_label)
cell_label.FillComponent(0,0.0)
#Label Assignment
for ig in range(ncells):
  mesh.GetCellPoints(ig,ptlst)
  mesh.GetPoints().GetPoints(ptlst,pt)
  barycenter(pt,pb)
  pt.Reset()
  im=ploc.FindClosestPoint(pb)
  cell_label.SetTuple1(ig,iml.GetComponent(im,0))
writer = vtk.vtkUnstructuredGridWriter()
writer.SetFileName("Mesh_adapt_final.vtk")
writer.SetInput(mesh)
writer.Update()
s=str(ncells)
#Creation of the diffusion file
f = open('Diff', 'w')
f.write(s+'\n')
for z in range(ncells):
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(z,0)
                                                      ==20):
    f.write('0.0013\n')
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(z,0)
                                                      ==50):
    f.write('0.0065\n')
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(z,0)
                                                      ==10):
    f.write('0.1000\n')
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(z,0)
                                                      ==0):
    f.write('0.\n')
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(z,0)
                                                      ==100):
    f.write('0.\n')
f.close()
#Creation of the initial condition
c = open('C0', 'w')
c.write(s+'\n')
for zc in range(ncells):
  if(mesh.GetCellData().GetArray("cell_labels").GetComponent(zc,0)
                                                       ==100):
    c.write('1.\n')
  else:
    c.write('0.\n')
c.close()
```

A.2 *FreeFem++* code

A.2.1 Linear Diffusion Case

Main Code

```
//Linear.edp
verbosity=1;
load "iovtk"
int[int] ffordervel=[0];
mesh3 Th;
Th=vtkload 3("Mesh_adapt_final.vtk");
int cd;
int cc;
real cpu=clock();
real ciclo;
include "Timing.idp"
include "FEspace_lin.idp"
include "Param.idp"
cpu=clock();
ciff=c0FE[];
diff=D[];
for(cd=0; cd<dimYh; cd++)
{
  if(ciff[cd]>=1)
    diff[cd]=0.;
}
D[]=diff;
time("DIFFUSION CORRECTION");
include "Variational_lin.idp"
Xh uno=1.;
real VT=int3d(Th)(uno);
c=c0;
set(C,solver=GMRES);
for (t=1;t<=T;t+=dt){
  cout << "STEP= " << t << ";" <<endl;
  ciclo=clock();
  cold=c;
  cpu=clock();
  rhs = varrhs(0,Xh);
  time("RHS");
  cpu=clock();
  C=varcc(Xh,Xh);
  c[]=C^-1*rhs;
  time("RISOLUZIONE SISTEMA");
  cpu=clock();
  cxD=c;
  ciff=cxD[];
  diff=D[];
  for(cd=0; cd<dimYh; cd++)
  {
    if(ciff[cd]>=1)
      diff[cd]=0.;
  }
  D[]=diff;
  time("DIFFUSION")
  timeCiclo;
  cout << "VOLUME: " << int3d(Th)(volume*(c>1e-4))/VT << endl;
  if(t==195)savevtk("Tumor1000_"+e11+e22+e33+"_newmap_"+t+".vtk",
                          Th,c,order=ffordervel,dataname="Tumor");
}
savevtk("Tumor1000_"+e11+e22+e33+"_newmap_"+T+".vtk",
                          Th,c,order=ffordervel,dataname="Tumor");
```

FE Space Code

```
//FEspace_lin.idp
fespace Xh(Th,P23d);
fespace Yh(Th,P03d);
Xh c,cold,k,erp;
Yh D,cOFE,cxD;
int dimXh=Xh.ndof;
int dimYh=Yh.ndof;
real [int] corr(dimXh);
real [int] ciff(dimYh);
real [int] diff(dimYh);
```

Varational Forms Code

```
//Variational_lin.idp
varf varcc(c,k) = int3d(Th)(c*k/dt - rho*(1.-cold)*c*k + (D *
      [e11*dx(c)+e12*dy(c)+e13*dz(c),
                     e21*dx(c)+e22*dy(c)+e23*dz(c),
                     e31*dx(c)+e32*dy(c)+e33*dz(c)] ) * Grad(k));
varf varrhs(c,k) = int3d(Th) (cold*k/dt);
cpu=clock();
real [int] rhs(dimXh);
```

A.2.2 Non Linear Diffusion Case

Main Code

```
//Nonlinear.edp
verbosity=1;
load "iovtk"
int[int] ffordervel=[0];
mesh3 Th;
Th=vtkload 3("Mesh_adapt_final.vtk");
int cd;
int cc;
real cpu=clock();
real fixtime=clock();
real ciclo;
real tol=1e-6;
int maxit=1000;
int i;
real err;
include "Timing.idp"
include "FEspace_nl.idp"
include "Param.idp"
```

```
cpu=clock();
ciff=c0FE[];
diff=D[];
for(cd=0; cd<dimYh; cd++)
{
  if(ciff[cd]>=1)
    diff[cd]=0.;
}
D[]=diff;
time("DIFFUSION CORRECTION");
include "Variational_nl.idp"
Xh uno=1.;
real VT=int3d(Th)(uno);
c=c0;
for (t=1;t<=T;t+=dt){
  cout << "STEP= " << t << ";" <<endl;
  ciclo=clock();
  cold=c;
  cpu=clock();
  rhs = varrhs(0,Xh);
  time("RHS");
  cpu=clock();
  C=varcc(Xh,Xh);
  time("C EXTRACTION");
  int i=0;
  real err=1;
  fixtime=clock();
  while(i<=maxit && err>tol)
  {
    cpu=clock();
    Cnl=varf ix(Xh,Xh);
    time("Cnl");
    Ct=C+Cnl;
    set(Ct,solver=GMRES);
    c[]=Ct^-1*rhs;
    time("SYSTEM RESOLUTION");
    erp=c-cold;
    err=sqrt(int3d(Th)((erp)^2));
    cout << "L2 ERROR: " << err << endl;
    cold=c;
    i++;
  }
  timef;
  cpu=clock();
  cxD=c;
  ciff=cxD[];
  diff=D[];
  for(cd=0; cd<dimYh; cd++)
  {
      if(ciff[cd]>=1)
         diff[cd]=0.;
  }
  D[]=diff;
  time("DIFFUSION");
  corr=c[];
  for(int bc=0;bc<dimXh;bc++)
  {
    if(corr[bc]< 1e-5)
      corr[bc]=0;
  }
  c[]=corr;
  timeCiclo;
  cout << "VOLUME :" << " " << int3d(Th)(volume*(c>1e-4))/VT << endl;
  if(t==195)savevtk("Tumor1000_"+e11+e22+e33+"_nl_newmap_"+t+".vtk",
                          Th,c,order=ffordervel,dataname="Tumor");
}
savevtk("Tumor1000_"+e11+e22+e33+"_nl_newmap_"+T+".vtk",
                          Th,c,order=ffordervel,dataname="Tumor");
```

FE Space Code

```
//FEspace_nl.idp
fespace Xh(Th,P23d);
fespace Yh(Th,P03d);
Xh c,cold,k,erp;
Yh D,cOFE,cxD;
int dimXh=Xh.ndof;
int dimYh=Yh.ndof;
real [int] corr(dimXh);
real [int] ciff(dimYh);
real [int] diff(dimYh);
```

Varational Forms Code

```
//Variational_nl.idp
fespace Xh(Th,P23d);
fespace Yh(Th,P03d);
Xh c,cold,k,erp;
Yh D,cOFE,cxD;
int dimXh=Xh.ndof;
int dimYh=Yh.ndof;
real [int] corr(dimXh);
real [int] ciff(dimYh);
real [int] diff(dimYh);
varf varcc(c,k)=int3d(Th)(c*k/dt - rho*(1-cold)*c*k );
varf varf ix(c,k) = int3d(Th)( (2*cold*(D *
      [e11*dx(c)+e12*dy(c)+e13*dz(c),
                     e21*dx(c)+e22*dy(c)+e23*dz(c),
                      e31*dx(c)+e32*dy(c)+e33*dz(c)]) ) * Grad(k));
varf varrhs(c,k) = int3d(Th) (cold*k/dt);
matrix C;
matrix Cnl;
matrix Ct;
real [int] rhs(dimXh);
```

A.2.3 Other Include Files

Model Parameters

```
//Param.idp
//Diffusion Function
ifstream dfile("Diff");
```

```
dfile >> D[];
//Initial Condition
ifstream cfile("C0");
cfile >> c0FE[];
func c0=c0FE;
macro Grad(c) [dx(c),dy(c),dz(c)] //EOM
real t;
real dt=1;//day
real T=492;//days
real e11=1;
real e12=0;
real e13=0;
real e21=0;
real e22=1;
real e23=0;
real e31=0;
real e32=0;
real e33=1;
real rho=0.012;//day^-1
```

Time Calculation Code

```
//Timing.idp
macro time(name)
{
cout << name << ": " << clock() - cpu << " s." << endl;
cpu=clock();
}//
macro timeCiclo()
{
cout << "CICLE TIME: " << clock() - ciclo << " s." << endl;
}//
macro timef()
{
cout << i << " FIXED POINT ITERATIONS: " << clock() - fixtime << " ↩
   s." << endl;
fixtime=clock();
}//
```

Appendix B

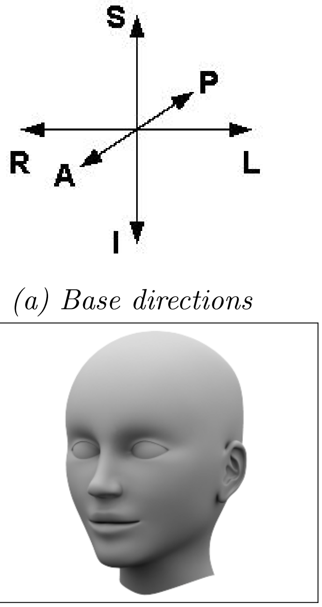
MhdOrientation

B.1 Medical Images

B.1.1 Orientation of the Images

Working on MR's output data is mandatory to be familiar with the concepts of orientation and base direction. The base directions are:

- **R** = right;
- **L** = left;
- **A** = anterior;
- **P** = posterior;
- **S** = superior;



(a) Base directions

(b) Patient

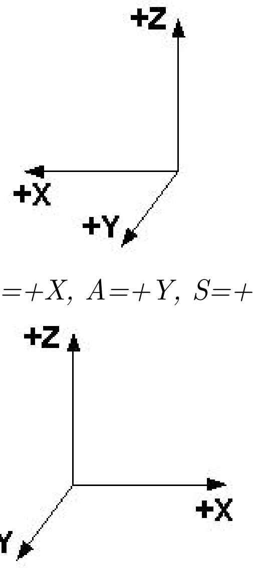
Figure B.1: Patient coordinate system.

- **I** = inferior;

indicating the axis using as reference the patient, see Figure [B.1]. These base directions could be combined in different manners in order to create a tern of axis, the letter related to a base direction indicates a positive semi–axis of the tern, see Figure [B.2]. The so generated tern could be left–handed ([B.2b]) or rigth–handed ([B.2a]) and defines the anatomical orientation of the image.

B.1.2 Problems in Working with Medical Images

We consider images formatted as *MetaImage*. They consist of two files: a .mhd file (a header file containing parameters used to read the image), and a .raw/.zraw



(a) R=+X, A=+Y, S=+Z.

(b) L=+X, A=+Y e S=+Z.

Figure B.2: Terns indicated by base directions.

file containing the data. In the header file is stored the information about the orientation, the rotation matrix and the offset.

Usually in biological process modeling, the interest is to obtain from a given medical image a proper computational mesh. Once the mesh is created, it is used for the numerical simulations and at the end of this process the results are analyzed comparing the mesh and the image by superposing them. Unfortunately during the process of mesh generation is possible to loss the information about the original position of the medical image.

This problem is partially solved using *vmtk* to create the mesh basing on a *MetaImage*, indeed it includes routines that allows to move the grid obtained on the starting image, if it is RAS oriented.

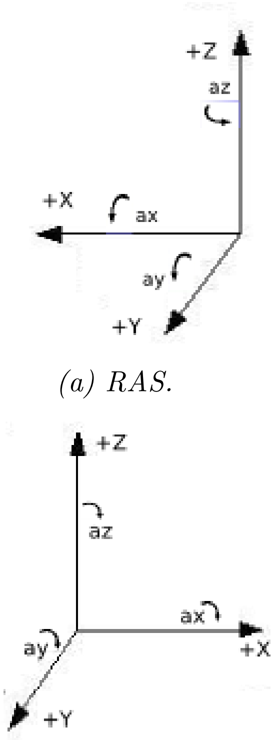
The aim of the library implemented is to solve the problem executing an image orientation's conversion from a generic anatomical orientation to a RAS orientation. In this way the process of mesh generation can be performed on a RAS oriented image and via the *vmtk*'s routines is possible to superpose the mesh on the image without loss of geometrical information.

B.1.3 Conversion Algorithm

The conversion is performed working on the header file of the image. The aim is to obtain a new header file that allows to read the image in a RAS coordinate system. First we calculate the Euler angles from the rotation matrix with the algorithm proposed in [Sla08], after that the angles and the offset are converted in to a RAS system. From the converted angles and the converted offset is possible to write a new header file that allows to read the image RAS oriented. For example in the case LAS→RAS (see Figure [B.3]) the conversion is

- X_{RAS}=$-X_{LAS}$;
- Y_{RAS}=Y_{LAS};
- Z_{RAS}=Z_{LAS};
- $\alpha_{X_{RAS}}$=$\alpha_{X_{LAS}}$;
- $\alpha_{Y_{RAS}}$=$-\alpha_{Y_{LAS}}$;
- $\alpha_{Z_{RAS}}$=$-\alpha_{Z_{LAS}}$.

Working similarly is possible to obtain a conversion to RAS for generic anatomical orientation. Following orientations are included in the library



(a) RAS.

(b) LAS.

Figure B.3: Orientation conversion from LAS to RAS.

- AIL→RAS;
- ASL→RAS;
- LAS→RAS;
- RAI→RAS.

B.2 Library Implementation

It is possible to download and install the library via *github* by typing:

```
git clone https://github.com/drugilsberg/mhdorientation
cd mhdorientation/
mkdir lib/build
make dynamiclib
```

The library is divided in two parts: a C++ version and a Python version (realized interfacing C++ code with Python).

B.2.1 C++ version

This part of the library is implemented using the factory paradigm. The classes used to implement the structure of the factory are:

- `MhdOrientation`:
  - `AIL`;

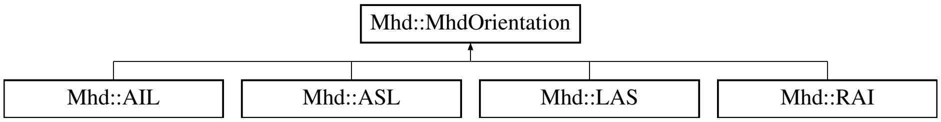


Figure B.4: Relationship between the base class and its derived classes.

  - `ASL`;
  - `LAS`;
  - `RAI`;
- `MhdFactory`;
- `MhdProxy`.

`MhdOrientation` and its Derived Classes

The class `MhdOrientation` is used to manage the image orientation and execute, with its derived classes, the conversion from a generic anatomical orientation to a RAS coordinate system. The relationship between these classes is described in Figure [B.4]. The parameters read from the header file and the Euler angles related to the rotation matrix are defined as `protected` members:

```
protected:
     vector<float> TransformMatrix;
     vector<float> Offset;
     vector<float> CenterOfRotation;
     string AnatomicalOrientation;
             .
```

```
              .
              .
              .
     pair< vector<float>, vector<float> > Angles;
```

The methods used to read and write the header file, the virtual conversion method and the virtual constructor are `public` members of the class.
The virtual methods are:

```
virtual void ConvertToRas(size_t i=1)=0;
virtual MhdOrientation * Create() const =0;
```

These methods are implemented in the derived classes. See *Doxygen* generated documentation in `doc/` for further details.

`MhdFactory`

This class is implemented as a *singleton* declaring as `private` members the constructors and the assignment operator, and delegating the construction of the class to the `public` method `Instance`:

```
MhdFactory & MhdFactory::Instance() {
    static MhdFactory theFactory;
     return theFactory;
    }
```

The two following typedefs are used to ease syntax:

```
typedef std::unique_ptr<MhdOrientation > (* MhdBuilder)();
typedef map<string,MhdBuilder> Collector;
```

A std::unique_ptr is used to avoid memory leaks. Collector is the map used to store the orientation rules builder.

The method Register:

```
void Register(string const & Name, MhdBuilder const & Func) ↩
   throw(invalid_argument);
```

is used to add new orientations to the factory, while the method Get:

```
unique_ptr<MhdOrientation> Get(string const & Name) const;
```

is used to extract an object MhdOrientation from the factory. For details see the documentation.

MhdProxy

This class is used to allow new orientation's automatic registration in the factory. Here the constructor and the Build method are reported:

```
template<typename T>
   unique_ptr<MhdOrientation> MhdProxy<T>::Build() {
     return unique_ptr<MhdOrientation>(new T);
   }

template<typename T>
   MhdProxy<T>::MhdProxy(char const * const & Name) {
     MhdFactory & factory(MhdFactory::Instance());
     factory.Register(string(Name),MhdProxy<T>::Build);
   }
```

It is important to notice the use of a template parameter to choose the orientation added.

The orientations already implemented are registered in order to be used when the factory is loaded dynamically:

```
namespace
{
  using namespace Mhd;
  //Registration of the orientation type
  MhdProxy<AIL> OA("AIL");
  MhdProxy<ASL> OB("ASL");
  MhdProxy<RAI> OC("RAI");
  MhdProxy<LAS> OD("LAS");
}
```

Using the method `Get` described before is possible to extract one of them from the factory. See the documentation for details.

B.2.2 Python Interface

The interface is implemented using the class `MhdOrientationPython`. This class is similar to `MhdOrientation` but has a more primitive structure. Since it is a concrete object no virtual methods are declared and all the steps to perform the conversion are `public` members of the class and are directly implemented. The parameters and the Euler angles are `private` members. For implementation details see the documentation.

We define in an `extern "C’"` enviroment a list of functions that re–implement the methods and the constructor of the class `MhdPythonOrientation`:

```
#include "MhdPythonOrientation.hxx"

extern "C" {

 Mhd::MhdPythonOrientation *MhdOrientation_Python(){return new ↩
    Mhd::MhdPythonOrientation();}

 void MhdOrientation_OrientationReader(Mhd::MhdPythonOrientation* ↩
    mo, char* InputFile) {mo->OrientationReader(InputFile);}

 void MhdOrientation_OrientationWriter(Mhd::MhdPythonOrientation* ↩
    mo, char* OutputFile) {mo->OrientationWriter(OutputFile);}

 void MhdOrientation_ConvertToRas(Mhd::MhdPythonOrientation* ↩
    mo,size_t i) {mo->ConvertToRas(i);}
```

```
  void MhdOrientation_ComputeAngles(Mhd::MhdPythonOrientation* mo) ↩
      {mo->ComputeAngles();}

  void MhdOrientation_ComputeRotation(Mhd::MhdPythonOrientation* mo, ↩
      float* angles){mo->ComputeRotation(angles);}

  void MhdOrientation_R(Mhd::MhdPythonOrientation* mo,size_t i,size_t ↩
      j){mo->R(i,j);}

  void MhdOrientation_O(Mhd::MhdPythonOrientation* mo,size_t ↩
      i){mo->O(i);}

  void MhdOrientation_C(Mhd::MhdPythonOrientation* mo,size_t ↩
      i){mo->C(i);}

  void MhdOrientation_AO(Mhd::MhdPythonOrientation* mo){mo->AO();}
}
```

Now is possible to export them in Python, loading the library with *ctypes*, and use this functions to create a Python class as follows:

```
#Module mhd.py
import ctypes
lib = ctypes.CDLL('./libMhdOrientation.so',mode=ctypes.RTLD_GLOBAL)
```

```
class MhdOrientation(object):

  def __init__(self):
    self.obj = lib.MhdOrientation_Python()


  def OrientationReader(self,InputFile):
    lib.MhdOrientation_OrientationReader(self.obj,InputFile)


  def OrientationWriter(self,OutputFile):
    lib.MhdOrientation_OrientationWriter(self.obj,OutputFile)


  def ConvertToRas(self,i=1):
    lib.MhdOrientation_ConvertToRas(self.obj,i)


  def ComputeAngles(self):
    lib.MhdOrientation_ComputeAngles(self.obj)


  def ComputeRotation(self,angles):
    lib.MhdOrientation_ComputeRotation(self.obj,angles)
```

```
def R(self,i,j):
  lib.MhdOrientation_R(self.obj,i,j)


def O(self,i):
  lib.MhdOrientation_O(self.obj,i)


def C(self,i):
  lib.MhdOrientation_C(self.obj,i)


def AO(self):
  lib.MhdOrientation_AO(self.obj)
```

The module `mhd` can be imported in Python and allows to use the library in a more user friendly way, making its interaction with other software (e.g. *vmtk*), easier. For a complete description of the library see the documentation.

Bibliography

[AJ91] Ellsworth C. Alvord Jr., *Simple model of recurrent gliomas.*, J Neurosurg **75** (1991), 337–338.

[AJ94a] ______, *The mathematical relationships among various proliferation indexes.*, Am J Surg Pathol **18** (1994), 856–857.

[AJ94b] ______, *Survival in malignant gliomas.*, J Neurosurg **80** (1994), 350–351.

[AJ95] ______, *Patterns of growth of gliomas.*, AJNR Am J Neuroradiol **16** (1995), 1013–1017.

[AJS91] Ellsworth C. Alvord Jr. and C.M. Shaw, *Neoplasms affecting the nervous system in the elderly.*, In: Duckett S, ed. The Pathology of the Aging Human Nervous System. (1991), 210–281.

[BKMAJ97] P.K. Burgess, P.M. Kulesa, J.D. Murray, and E.C. Alvord Jr., *The interaction of growth rates and diffusion coefficients in a three-dimensional mathematical model of gliomas.*, J Neuropathol Exp Neurol **56** (1997), 704–713.

[CLT56] P.V. Collins, R.K. Loeffler, and H. Tivey, *Observations on growth rates of human tumors*, Am J Roentgenol Radium Ther Nucl Med **76** (1956), 988–100.

[GM77] M. Gurtin and Richard C. Maccamy, *On the diffusion of biological populations*, Mathematical Biosciences **33** (1977), no. 1-2, 35–49.

[HAJS07] H.L.P. Harpold, Ellsworth C. Alvord Jr., and K.R. Swanson, *The evolution of mathematical modeling of glioma proliferation and invasion*, Journal of Neuropathology & Experimental Neurology **66** (2007), no. 1, 1–9.

[HTC01] S.M. Haney, P.M. Thompson, and et al. Cloughesy, T.S., *Tracking tumor growth rates in patients with malignant gliomas: A test of two algorithms.*, AJNR Am J Neuroradiol **22** (2001), 73–82.

[JMD$^+$05] S. Jbabdi, E. Mandonnet, H. Duffau, L. Capelle, K. R. Swanson, M. Pélégrini-Issac, R. Guillevin, and H. Benali, *Simulation of anisotropic growth of low-grade gliomas using diffusion tensor imaging*, Magnetic Resonance in Medicine **54** (2005), no. 3, 616–624.

[Mur03] J.D. Murray, *Mathematical biology II: Spatial models and biomedical applications*, Intercisciplinary Applied Mathematics: Mathematical Biology, Springer, 2003.

[Qua08] A. Quarteroni, *Modellistica numerica per problemi differenziali*, UNITEXT / La Matematica per il 3+2, Springer, 2008.

[SAJ02] K.R. Swanson and Ellsworth C Alvord Jr., *A biomathematical and* *pathological analysis of an untreated glioblastoma.*, 7th European Congress of Neuropathology (Helsinki, Finland) (2002), 28.

[SAJM92] K.R. Swanson, Ellsworth C. Alvord Jr., and J.D. Murray, *A quantitative model for differential motility of gliomas in grey and white matter.*, Cell Prolif (1992), 32–48.

[SAJM02a] ______, *Quantifying efficacy of chemotherapy of brain tumors (gliomas) with homogeneous and heterogeneous drug delivery.*, Acta Biotheor **50** (2002), 223–237.

[SAJM02b] ______, *Virtual brain tumors (gliomas) enhance the reality of medical imaging and highlight inadequacies of current therapy.*, Br J Can **86** (2002), 14–18.

[SAJM03] ______, *Virtual resection of gliomas: effects of location and extent of resection on recurrence.*, Math Comput Model (2003).

[Sal] S. Salsa, *Equazioni a derivate parziali: Metodi, modelli e applicazioni*, Springer.

[SBMAJ03] K.R. Swanson, C. Bridge, J.D. Murray, and Ellsworth C. Alvord Jr., *Virtual and real brain tumors: using mathematical modeling to quantify glioma growth and invasion*, Journal of the neurological sciences **216** (2003), no. 1, 1–10.

[Sla08] Gregory G. Slabaugh, *Computing Euler angles from a rotation matrix*, Tech. report, 2008.

[Ste77] G.G. Steel, *Growth kinetics of tumours. cell population kinetics in relation to the growth and treatment of cancer.*, Oxford: Clarendon Press (1977).

[Swa99] K.R. Swanson, *Mathematical modeling of the growth and control of tumors.*, PhD thesis. University of Washington (1999).

[TCW+95] P. Tracqui, G.C. Cruywagen, D.E. Woodward, G.T. Bartoo, J.D. Murray, and E.C. Alvord Jr., *A mathematical model of glioma growth: the effect of chemotherapy on spatio-temporal growth.*, Cell Prolif **28** (1995), 17–31.

[TCW+96] P. Tracqui, G.C. Cruywagen, D.E. Woodward, J. Cook, Murray J.D., and E.C. Alvord Jr, *A mathematical model of glioma growth: the effect of extent of surgical resection.*, Cell Prolif **29** (1996), 269–288.

[Vaz07] J.L. Vazquez, *The Porous Medium Equation, Mathematical Theory*, Claredon Press,Oxford, 2007.

[Vaz12] ______, *Barenblatt solutions and asymptotic behaviour for a nonlinear fractional heat equation of porous medium type*, ArXiv e-prints (2012).

[Wil92] C.B. Wilson, *Glioblastoma: The past, the present, and the future.*, Clin Neurosurg **38** (1992), 32–48.